The deMon User's Guide

Installation Guide and Reference Manual

The deMon User's Guide
Installation Guide and Reference Manual

Version 1.0.3
June 2004

Reference: The deMon User's Guide, Version 1.0.3, 2003-2004

Authors: A.M. Köster, P. Calaminici, S. Escalante, R. Flores-Moreno, A. Goursot, S. Patchkovskii, J.U. Reveles, D.R. Salahub, A. Vela, T. Heine

Title Picture: Correlation Diagram for the Ionization of V_3

The procedures and applications presented in this guide have been included for their instructional value. They have been tested with care but are not guaranteed for any particular purpose. The authors do not offer any warranties or representations, nor do they accept any liabilities with respect to the programs or applications.
The software described in this guide is furnished under a license and may be used or copied only in accordance with the terms of such license.

Copyright ©2003-2004 by the authors
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without prior written permission of one of the authors.

Contents

1 Getting Acquainted **1**
1.1 The Game of the Name . . . 1
1.2 How to Avoid Reading this Manual . . . 2
1.3 How to Read this Manual . . . 2
1.4 How to Use deMon . . . 3

2 Getting Started **7**
2.1 Before you Begin . . . 7
2.2 How to Install deMon . . . 7
2.2.1 Contents of the Distribution Package . . . 7
2.2.2 Installation Prereqisites . . . 8
2.2.3 Express Installation . . . 9
2.2.4 Advanced Installation . . . 9
2.3 Porting deMon to a new platform . . . 11
2.4 Tuning deMon for a specific host . . . 12
2.5 What to do when things go wrong . . . 13
2.6 How to Run deMon . . . 16

3 Carrying On **19**
3.1 Parameters . . . 19
3.2 Files . . . 19
3.3 Input Syntax . . . 20

4 Keywords **23**
4.1 Geometry Input . . . 23
4.1.1 Keyword **GEOME**TRY . . . 23
4.1.2 Keyword **CONST**ANTS . . . 32
4.1.3 Keyword **VARIA**BLES . . . 33
4.1.4 Keyword **SYMME**TRY . . . 33
4.2 Basis Set Input . . . 35
4.2.1 Keyword **BASIS** . . . 35
4.2.2 Keyword **AUXIS** . . . 36
4.2.3 Keyword **ECPS** . . . 38

4.2.4 Keyword **MCPS** . . . 39
4.3 Electronic State Control . . . 40
4.3.1 Keyword **MULTI**PLICITY . . . 40
4.3.2 Keyword **CHARG**E . . . 40
4.3.3 Keyword **MOEXC**HANGE . . . 40
4.3.4 Keyword **FIXMO**S . . . 41
4.3.5 Keyword **SMEAR** . . . 41
4.3.6 Keyword **CONFI**GURE . . . 42
4.4 SCF Control . . . 46
4.4.1 Keyword **SCFTY**PE . . . 46
4.4.2 Keyword **ORBIT**ALS . . . 47
4.4.3 Keyword **ERIS** . . . 48
4.4.4 Keyword **GUESS** . . . 50
4.4.5 Keyword **MIXIN**G . . . 51
4.4.6 Keyword **SHIFT** . . . 52
4.4.7 Keyword **DIIS** . . . 53
4.4.8 Keyword **VXCTY**PE . . . 54
4.4.9 Keyword **GRID** . . . 55
4.5 Molecular mechanics and QM/MM Control . . . 58
4.5.1 Keyword **QMMM** . . . 58
4.5.2 Keyword **FORCE**FIELD . . . 59
4.5.3 Keyword **MMOPT**IONS . . . 60
4.5.4 Keyword **VDWAA**LS . . . 60
4.5.5 Keyword **MADEL**UNG . . . 61
4.5.6 Keyword **PEDIO**IC . . . 61
4.6 Optimization Control . . . 63
4.6.1 Keyword **OPTIM**IZATION . . . 63
4.6.2 Keyword **HESSI**AN . . . 64
4.6.3 Keyword **STEPT**YPE . . . 65
4.6.4 Keyword **UPDAT**E . . . 66
4.7 MD Control . . . 67
4.7.1 Keyword **MDYNA**MICS . . . 67
4.7.2 Keyword **MDSTE**PS . . . 68

4.7.3 Keyword **MDTEM**P 68
4.7.4 Keyword **TIMES**TEP 69
4.7.5 Keyword **MDBAT**H 69
4.7.6 Keyword **CONSE**RVE 71
4.7.7 Keyword **MDCON**STRAINTS 72
4.7.8 Keyword **MDPRE**SSURE 72
4.8 Property Control 74
4.8.1 Keyword **POPUL**ATION 74
4.8.2 Keyword **DIPOL**E 74
4.8.3 Keyword **MAGOU**T 74
4.8.4 Keyword **POLAR**IZABILITY 75
4.8.5 Keyword **FREQU**ENCY 76
4.8.6 Keyword **THERM**O 77
4.8.7 Keyword **FNMC** 77
4.8.8 Keyword **HARDNESS** 77
4.9 Visualization and Topology 79
4.9.1 Keyword **VISU**ALIZATION 79
4.9.2 Keyword **PLOT** 80
4.9.3 Keyword **CPSEA**RCH 82
4.9.4 Keyword **ISOSU**RFACE 83
4.9.5 Keyword **GEOSU**RFACE 86
4.9.6 Keyword **BOX** 87
4.9.7 Keyword **POINT**S 89
4.10 Miscellaneous Keywords 91
4.10.1 Keyword **MAXMEM** 91
4.10.2 Keyword **TITLE** 91
4.10.3 Keyword **PRINT** 91
4.10.4 Keyword **EMBED** 93
4.10.5 Keyword **CHOLE**SKY 94
4.10.6 Keyword **MATDI**A 94
4.10.7 Keyword **WEIGH**TING 95
4.10.8 Keyword **QUADR**ATURE 96
4.10.9 Keyword **ECPIN**TEGRATION 96

4.10.10 Keyword **GENAO** 96

5 Examples **98**
5.1 Example ps18 98
5.2 Examples amk5, amk6, amk7, and amk8 98
5.3 Examples amk29, amk30, and amk31 98
5.4 Example ps86 98
5.5 Example th6 98
5.6 Example amk17 98
5.7 Example ps84 98
5.8 Example amk19 98
5.9 Example amk20 98
5.10 Example ps15 98
5.11 Example ps6 98
5.12 Example amk24 98
5.13 Example amk26 98
5.14 Example amk33 98
5.15 Example amk35 98
5.16 Example ps88 98
5.17 Example ps90 98

6 Using QM/MM in deMon **99**

7 Using MD in deMon **100**

8 Vu **101**
8.1 What is Vu? 101
8.2 Vu and deMon 102
8.3 Running Vu 102

9 MAG **104**

10 Quick Keyword Reference **105**

11 Troubleshooting **106**

A Automatic Generation of Auxiliary Functions **107**

B Format of a platform description file **109**

B.1 Philosophy of deMon configuration files 109

B.2 Structure of a platform configuration files 109

C Working with the deMon Test Suite **115**

D Global Counters, Limits and Pointers **116**

References **119**

List of Tables

1 Platform-specific installation notes 7
2 Parameter Settings 20
3 Logical and Physical Filenames 21
4 Special Input Symbols 21
5 Internal Coordinate Substitution 27
6 Basis Sets 37
7 ERI Calculation 49
8 Exchange and Correlation Functionals 56
9 Molecular Fields 82
10 Structure of a platform configuration file 110
11 Platform Specific Functions 114
12 Counters 116
13 Limits 117
14 Pointers 118

List of Figures

1 Z-Matrix Input Orientation 25
2 Dihedral Angle Definition 26
3 Mixed Input Definition 29

1 Getting Acquainted

1.1 The Game of the Name

deMon is a system of programs for density functional theory (DFT) calculations of atoms, molecules and solids [1–3]. Its first widely available version [4] appeared in 1992. Shortly after its appearance the original deMon code was substantially modified for commercialization by BIOSYM Technologies. The beta-release of this version appeared 1993. It was the basis of the deMon-KS1 [5] series of programs developed in Montreal until 1997.
Meanwhile the original deMon version was further developed in Montpellier and Stockholm. These developments were first independent from each other. In 1997 they merged to the deMon-KS3 [6] series of programs.
Independently from the deMon development the AllChem project started [7] in Hannover in 1995. The aim of this project was to write a well structured DFT code from scratch. The first AllChem version appeared in 1997. The structured programming of AllChem proved very useful for the development and testing of new DFT approaches and algorithms.
In March 2000 the first deMon developers meeting was hold in Ottawa. At this meeting the deMon and AllChem developers agreed to merge their codes in order to keep a Tower of Babel from rising. As a result the new code couples the deMon functionality with the stable and efficient integral and self-consistent field (SCF) part from AllChem. The new code, which is named deMon 2003 [8], was presented the first time at the third deMon developers meeting in Geneva.
The new deMon 2003 code is available for just about any computer made today, including parallel architectures. The versions for the different platforms are essentially the same. The input file created according to this guide should give, within the numerical accuracy of the used compile flags, the same output.
The written word carries more legal complications than the spoken, and the need to distinguish deMon 2003 from similarly spelled products restricts how you may write it. The best way to refer to the program in the text is to use the unusual capitalization. The new deMon code has to be cited as:

A.M. Köster, R. Flores-Moreno, G. Geudtner, A. Goursot, T. Heine,
J.U. Reveles, A. Vela, S. Patchkovskii, D.R. Salahub, deMon 2004, NRC, Canada

There is another companion to this guide, *The deMon Programmer's Guide*, which cur-

rently under development. This companion is an in-depth guide about the programming philosophy and style of the new deMon code. Its the place to look if you want to extend or modify the program. To learn more about other deMon utility programs and interfaces as well as activities of the deMon user community look at *http://www.deMon-software.com*.

1.2 How to Avoid Reading this Manual

If you are familiar with an older version of deMon or you rather prefer to learn by example, look at the directory `examples` in the deMon tree you have obtained. It contains sample inputs for the most common applications (see Chapter 5 for the description of the examples). In Chapter 10 you will find a quick reference guide of the input keywords. Common problems and their solutions are described in Chapter 11, *"Troubleshooting"*. If you prefer to learn more about the deMon input syntax before you use it, read on. Almost everything used in the sample inputs is explained in the next two chapters.

1.3 How to Read this Manual

While the example inputs cover the most common standard applications of an quantum chemistry program like deMon, there is much more to learn about input possibilities. Eventually, you will want to control specific parts of a calculation or you will need more output information. You will then have to look in this manual for the necessary information. You can read the information containing section without the preceding ones. However, all later chapters assume that you have read Chapters 2 and 3.
For example, suppose you want to print molecular orbitals (MOs). Looking up the table of contents will lead you to Section 4.10.3, which describes the PRINT Keyword. The concept of keywords and options, as well as their notation in this guide, is described in Chapter 3. It will take you just a moment to construct the input line for printing molecular orbitals if you have already read chapter 3; it could be quite frustrating if you haven't. So it's best to read the three first chapters now. Here is a brief sketch of what's in the remaining chapters:

Chapter 2 gives an overview about how to install and run the program using the makefiles and scripts which come with deMon.

Chapter 3 explains how to customize the program parameters and input/output files. It also explains the input syntax.

Chapter 4 gives a detailed description of all input keywords and their options.

Chapter 5 discusses the input and output of the examples.

Chapter 8 describes the interface between deMon and Vu, a configurable visualization tool. It also explains basic operations within Vu.

Chapter 9 describes the interface between deMon and MAG, a program for the calculation of NMR properties. It also describes the use of MAG.

Chapter 10 is a quick reference guide to all input keywords.

Chapter 6 is a more in-depth introduction to QM/MM features implemented in deMon.

Chapter 7 gives additional hints on running molecular dynamics simulations.

Chapter 11 describes troubleshooting of common problems.

Appendix A describes the automatic generation of auxiliary function sets.

Appendix B describes the format of the platform configuration files, used by the deMon build system.

Appendix C describes the structure test suite for deMon, and how to expand it.

1.4 How to Use deMon

In deMon linear combinations of atomic Gaussian–type orbitals (LCGTO) are used for representing Kohn-Sham orbitals. In this ansatz the Kohn–Sham orbitals $\psi_i(\mathbf{r})$ are given by:

$$\psi_i(\mathbf{r}) = \sum_{\mu} c_{\mu i}\, \mu(\mathbf{r}) \tag{1.1}$$

Here $\mu(\mathbf{r})$ represents an atomic orbital (build from contracted Gaussians) and $c_{\mu i}$ the corresponding molecular orbital coefficient. With this expansion we find for the electronic density:

$$\rho(\mathbf{r}) = \sum_{\mu,\nu} P_{\mu\nu}\, \mu(\mathbf{r})\, \nu(\mathbf{r}) \tag{1.2}$$

$P_{\mu\nu}$ represents an element of the (closed-shell) density matrix, defined as:

$$P_{\mu\nu} = 2 \sum_{i}^{occ} c_{\mu i}\, c_{\nu i} \tag{1.3}$$

Using the LCGTO expansions for the Kohn–Sham orbitals (1.1) and the electronic density (1.2), the Kohn–Sham self-consistent field (SCF) energy expression [9] can be calculated as:

$$E_{SCF} = \sum_{\mu,\nu} P_{\mu\nu}\, H_{\mu\nu} + \frac{1}{2} \sum_{\mu,\nu} \sum_{\sigma,\tau} P_{\mu\nu}\, P_{\sigma\tau} \langle\, \mu\nu \,\|\, \sigma\tau \,\rangle + E_{xc}[\rho] \tag{1.4}$$

The total energy is the sum of E_{SCF} and the nuclear repulsion energy, which can be calculated analytically. In (1.4), $H_{\mu\nu}$ represents matrix elements of the core Hamiltonian. They are built from the kinetic and nuclear attraction energy of the electrons and describe the movement of an electron in the nuclear framework. The second term in (1.4) represents the Coulomb repulsion energy of the electrons. In contrast to Hartree–Fock theory, the calculation of the Coulomb and exchange energies are separated in DFT. For the calculation of the exchange–correlation energy $E_{xc}[\rho]$ a numerical integration has to be performed. In deMon, the calculation of the N^4 scaling Coulomb repulsion energy is avoided by introducing an auxiliary function expansion for the electron density. This approximated density $\tilde{\rho}(\mathbf{r})$ is expanded in primitive Hermite Gaussians $\bar{k}(\mathbf{r})$ which are centered at the atoms:

$$\tilde{\rho}(\mathbf{r}) = \sum_{k} x_k\, \bar{k}(\mathbf{r}) \tag{1.5}$$

With the LCGTO expansion for $\rho(\mathbf{r})$ and $\tilde{\rho}(\mathbf{r})$ we obtain the following approximate SCF energy:

$$\begin{aligned} E_{SCF} \;=\; & \sum_{\mu,\nu} P_{\mu\nu}\, H_{\mu\nu} + \sum_{k} x_k \sum_{\mu,\nu} P_{\mu\nu} \langle\, \mu\nu \,\|\, k \,\rangle - \\ & \frac{1}{2} \sum_{k,l} x_k\, x_l \langle\, k \,\|\, l \,\rangle + E_{xc}[\rho] \end{aligned} \tag{1.6}$$

Therefore, only three–center electron repulsion integrals (ERIs) are necessary for the SCF and energy calculation in deMon. This represents the most accurate energy model available in deMon. It is activated by the keyword VXCTYPE BASIS (see Section 4.4.8 for more details about the VXCTYPE keyword). By default (VXCTYPE AUXIS), the approximated energy is also used for the calculation of the exchange–correlation energy:

$$\begin{aligned} E_{SCF} \;=\; & \sum_{\mu,\nu} P_{\mu\nu}\, H_{\mu\nu} + \sum_{k} x_k \sum_{\mu,\nu} P_{\mu\nu} \langle\, \mu\nu \,\|\, k \,\rangle - \\ & \frac{1}{2} \sum_{k,l} x_k\, x_l \langle\, k \,\|\, l \,\rangle + E_{xc}[\tilde{\rho}] \end{aligned} \tag{1.7}$$

This approximation has proven very accurate ($\sim$1 kcal/mol) in combination with GEN-A* auxiliary functions sets (see Section 4.2.2 and Appendix A for the selection and auto-

matic generation of GEN-A* auxiliary function sets). Because it represents a considerable saving in computational time we suggest to use this approximation at the beginning of each study. If high accuracy is requested the systems optimized with VXCTYPE AUXIS (1.7) should be further optimized with VXCTYPE BASIS (1.6). Usually, convergence will be achieved within a few cycles. If sensitive properties like frequencies, polarizabilities etc. are intended to be calculated, the use of VXCTYPE BASIS is recommended. The differences between VXCTYPE AUXIS and VXCTYPE BASIS for geometries and bond energies are in the range of the accuracy of the methodology. An exception should be made for very weak bonds, where the geometry may be significantly affected by the use of fitted density. It should be noted that the default setting for the auxiliary functions is A2, independent which energy expression is used (see Section 4.2.2). For all theoretical models available in deMon VXCTYPE AUXIS results can be used as a restart guess (GUESS RESTART; see Section 4.4.4) for VXCTYPE BASIS calculations.

The most often encountered problem in DFT calculations is the failure of the SCF convergence. This is usually due to the small gap between the highest occupied (HOMO) and lowest unoccupied (LUMO) molecular orbital. In deMon the DIIS procedure (Section 4.4.7) is activated by default. For a small HOMO–LUMO gap DIIS may be counterproductive and should be switched off. Several alternative methods are available in deMon to enforce SCF convergence. Most important are modification in the choice of the starting GUESS (Section 4.4.4) and the MIXING (Section 4.4.5) procedure of the old and new densities and the enlarging of the HOMO–LUMO gap by the level SHIFT (Section 4.4.6) procedure. In all cases it is recommended to check the orbital energies and occupations using the PRINT keyword (Section 4.10.3). Other relevant keywords to alter or enforce SCF convergence are MOEXCHANGE (Section 4.3.3), FIXMOS (Section 4.3.4) and SMEAR (Section 4.10.3). For atomic calculation the CONFIGURATION keyword (Section 4.3.6) should be used in order to ensure SCF convergence.

By default the electron repulsion integrals (ERIs) are calculated at the beginning of the SCF and kept in main memory (RAM). For larger systems the ERIs may not fit into the RAM. In this case deMon will automatically switch to a direct SCF procedure, which may also be activated manually, by using option DIRECT of the ERIS keyword (see 4.4.3). It is also possible to store ERIs on disk, by using ERIS CONVENTIONAL input keyword. However, this results in an I/O bottleneck, which will slow down the calculation on most computer architectures. For very large systems the option MULTIPOLE of the

ERIS keyword is recommended. It activates an asymptotic expansion of the ERIs, which results in a considerable time saving. It should also be noted that for very large systems the linear algebra steps in deMon may become a bottleneck. The keyword PRINT RAM may be useful in resolving such issues. The amount of memory required in the matrix diagonalization step (which dominates memory requirements for large systems) can be adjusted with the keyword MATDIA (see 4.10.6).
If the calculation has to be restarted the new input file produced in each deMon run may be useful. The input may be modified and extended. However, the molecular geometry specified in this file should not be changed if the restart file will be used. However, the new input file may change the ordering of the atoms in the molecule. As a result, it should **not** be used for continuing molecular dynamics simulations.

2 Getting Started

2.1 Before you Begin

Before you begin the installation process, please make sure to consult operating system specific instructions, which came with the distribution. They are found in the subdirectory "`doc/os`" of the deMon distribution. The list of system-specific notes, available at the time this manual was prepared, is given in table1.

Table 1: Platform-specific installation notes in `doc/os`.

README.forcheck	Forchek (http://www.forcheck.nl/)
README.ifc	Intel Fortran compiler on Linux/x86
README.pathscale	PathScale Fortran-90 compiler on Linux/x86-64 (AMD64)
README.pgi	Portland Group Fortran-90 compiler on Linux/x86 and Linux/x86-64
README.SunOS	SUN OS and Solaris hosts
README.Tru64	OSF/1, Digital Unix, and Tru64 hosts
README.windows	MicroSoft Windows

These notes summarise many days of porting and debugging, and may save you countless hours of frustration!

2.2 How to Install deMon

2.2.1 Contents of the Distribution Package

deMon distribution package contains the following:

basis/	-	deMon basis sets.
bin/	-	Executable directory
doc/	-	Documentation.
examples/	-	Some examples, and the test suite.
include/	-	Include files.
makefiles/	-	Make files (see section 2.2.4).
objects/	-	Scratch area for intermediate build objects.

source/	-	deMon source code.
timings/	-	Some representative timing data.
tools/	-	Additional deMon programs.
database/	-	Unsupported CREX configuration files
unsupported/	-	Various unsupported tools and scripts

Please keep in mind that many subdirectories contain specific "`README.txt`" files. You may find those useful.

2.2.2 Installation Prereqisites

You will need a computer system capable of running deMon. At the absolute minimum, you will need 1 Gbyte of free disk space to compile deMon, and run the installation test set. To run the full test set, you will need at least 512 Mbytes of RAM installed in your computer. Your computer must run a Unix-like operating system to install deMon successfully. (We may be able to supply precompiled binaries for non-Unix platforms).
You will also need a working Fortran-90 [1] compiler. Not all compilers claiming to be Fortran-90 actually are, or are capable of producing correct machine code. Unless deMon has already been tested with your compiler (see 2.1), you very likely *will* see compilation problems, ranging from compiler crashes, to run-time aborts, to incorrect results. On average, we discover 2-5 compiler bugs (and a few bugs in deMon itself) every time deMon gets ported to a completely new Fortran compiler. You have been warned.
Additionally, for best performance, you will need optimized BLAS libraries. If your vendor does not provide such, you will have to obtain and install one of the publicly available BLAS libraries yourself. A good place to start is: *http://www.netlib.org/atlas/*. Having an optimized LAPACK is also helpful, but less important.
If your BLAS and LAPACK are installed in non-standard directories, you can specify their location by setting the LIBPATH environment variable. For example, if your BLAS libraries are in the directory "`/usr/local/lib64`", you should say:

```
export LIBPATH="-L /usr/local/lib64/" # for bash
setenv LIBPATH "-L /usr/local/lib64/" # for csh
```

[1] Many people insist on spelling this language name as FORTRAN. In fact, the correct spelling is simply "Fortran". It is specified by an international standard (ISO/IEC 1539).

If you require an unusual command line option, or a linker parameter to make your BLAS work, it can be included in LIBPATH as well. This command has to be executed each time you build deMon - so you may find it useful to put it in your shell profile.

2.2.3 Express Installation

If deMon has been previously ported to the operating system and compiler you are using, and all needed libraries are installed in standard locations, you can use the express installation procedure, namely:

1. Unpack deMon distribution files in subdirectory "deMon" of your home directory.
2. Set deMon environment variable to point to the installation directory, for example:

```
export deMon=$HOME/deMon # For bash
setenv deMon $HOME/deMon # For csh
```

3. Run "install.sh" script:

```
cd $deMon
./install.sh
```

4. Follow instructions on your screen.

If no errors are reported, you have a working deMon installation. Otherwise, proceed to the advanced installation instructions in section 2.2.4.

2.2.4 Advanced Installation

The procedure outlined in this section is functionally identical to the express installation (2.2.3). However, you will have an opportunity to take immediate corrective action if any problems arise.

1. Set deMon environment variable to point to the installation directory, for example:

```
export deMon=$HOME/deMon # for bash
setenv deMon $HOME/deMon # for csh
```

   The rest of this section will assume, what deMon is installed in the directory $deMon.

2. If you intend to study very large systems, with more than 5000 atoms, or more than 10000 contracted basis functions, you will need to modify some of the settings in file "$deMon/include/parameter.h". See section 3.1 for the instructions.

3. Go to the $deMon/makefiles/ subdirectory, and issue the command:

   ```
   make makesys
   ```

   The installation script will attempt to find combinations of options, which are compatible with your platform. Please sevect the most specific set of parameters, which applies to your configuration. If the only platform choice you are given is "Generic settings", deMon does not have a pre-defined set of compilation options for your system. In this case, consult section 2.3 below. Please set the environment variable(s) suggested by the platform detection script. Usually, the only required variable is "`deMonPlatform`". To make these settings permanent, please add them to your shell initialization file (usually `.cshrc` or `.profile`, found in your home directory). The `deMonPlatform` setting will also affect the default choice of the deMon executable, selected by the deMon driver script (see Section 2.6).

4. It is usually a good idea to review the default compilation settings, particularly the library locations and optimization flags. The file `sys-$deMonPlatform.mak`, which is located in the directory `$deMon/makefiles/sys` contains this information.

   If you change any of the compilation options in the platform-specific makefile (`sys-$deMonPlatform.mak`), please execute the command "`make clean`" before rebuilding your program. Otherwise, your build may be inconsistent.

5. Build main deMon binary, by executing the command:

   ```
   make clean
   make build
   ```

   The first command (`make clean`) removes all old makefiles, objects, and binaries for this platform. The second command (`make build`) builds the main deMon executable, which is placed in the directory "`$deMon/bin/`".

   The build script takes care to keep all platform-specific data in unique files. It is therefore safe to perform several builds at the same time, as long as the `deMonPlatform` settings of the builds are different.

6. Build addional tools (MASTER and MAG), by issuing the command:

   ```
   make tools
   ```

   (You can combine the last two steps by using the command "`make clean build tools`").

7. Go to the $deMon/examples/ directory, and run deMon test suite:

   ```
   cd $deMon/examples
   make clean    # remove old test results
   make cheap    # execute minimal test set
   make all      # execute the complete test set
   ```

   (You can also say "`make clean all`", which is equivalent).

   The test suite (Appendix C) can take a while to run. If all goes right, you should see the message *"deMon installation tests completed successfully"* after each of the test phases. **Otherwise, do not use the deMon executable you have compiled.**

   Currently, the minimal test set requires about ten minutes on a mid-range PC (2.4GHz P4). The complete test set requires about two CPU-days. If you have a queuing system (such as PBS, SGE, or LSF) installed, you can run the test set in parallel. To activate parallel execution, set the environment variable `QSUB_COMMAND` to "`qsub`" (or whatever the name of your job submission command it):

   ```
   export QSUB_COMMAND=qsub # for bash
   setenv QSUB_COMMAND qsub # for csh
   ```

   before starting the test set. Please consult `$deMon/examples/README.txt` for more information on running deMon test set with batch queuing systems.

If the test set completes successfully, you are now ready to use deMon.

2.3 Porting deMon to a new platform

Compiling deMon on a previously upsupported platform involves a bit more work.

1. Examine the existing platform definition files in the "`$deMon/makefiles/sys/`" directory. Select a platform definition file, which is most similar to your system. If none of the platforms appear similar, choose "`sys-generic.mak`".

2. Create a copy of the prototype platform definition file, using a (short) descriptive name, corresponding to your platform.

3. Edit the new platform definiton file, following the instruction within the file itself. Also see the appendix B.

4. Set the environment variable deMonPlatform to reflect the name of your configuration file. For example, if your new platform configuration file is called "`sys-babayaga.mak`", the appropriate shell command is:

```
export deMonPlatform=babayaga # for bash
setenv deMonPlatform babayaga # for csh
```

5. Execute the commands:

```
cd $deMon/makefiles
make clean build tools
```

6. Run the test suite (see sections 2.2.4 and C).

7. If any of the tests fail, iterate to 3 above.

It would be appreciated if you can e-mail the platform definition to "`deMonkey@deMon-software.com`", for inclusion in future deMon distributions.

2.4 Tuning deMon for a specific host

You can almost always improve the preformance of deMon by using higher optimization levels, and/or supplying vendor-optimized mathematical libraries. The tuning procedure is very similar to the porting procedure outlined section 2.3 above. The only difference is:

1. In the directory `$deMon/makefiles/sys/`, choose a platform definition file which is already compatible with your system (If you do not remember which platforms are compatible, you can execute command `make makesys` in the directory `$deMon/makefiles`).

Hint: If you would like the platform file to apply to a single computer, the appropriate test in the `checkplatform:` rule is:

```
[ "`hostname`" = "whatever_the_name_is" ]
```

(Don't forget to start this line with a tab - or `make` will refuse to accept your configuration file).

Hint: You can choose the location to search for the optimized libraries by setting the LIBPATH environment variable, either in your shell initialization file, or in the platform configuration file itself.

2.5 What to do when things go wrong

First of all, Don't Panic. Installation of any complex software package is rarely entirely problem-free – and deMon consists of more than 1,200 source files and almost 200,000 lines of code.

There several ways of getting assistance. However, there are a few things you can do yourself before asking for help. First of all, please make sure that you are using the most up-to-date version of deMon – the problem you see may have already been resolved. Please also check that your Fortran compiler and numerical libraries are up-to-date. If this does not resolve your problem, here are a few more things you may want to try.

deMon build scripts rely on having a large subset of tools specified by the POSIX Unix standard (IEEE Std 1003.1-2001). If you need to build deMon on a platform lacking full POSIX support (such as Microsoft Windows or some exotic variants of Unix), you can do the following:

1. Find a POSIX-compliant Unix system. Linux or SGI Irix will work very well. Most recent Unix systems should also work.

2. Unpack deMon distribution, and follow advanced installation instructions in section 2.2.4. The platform you will need to build for is called "`flatsource`". It cannot be invoked through the express installation script. The build script will populate the `$deMon/flatsource` directory with symbolic links to all relevant source code files.

3. In the directory `$deMon/flatsource`, execute shell script `$deMon/unsupported/flat-makedep`
   It will create a rudimentary `Makefile` for building deMon.

4. Transfer all files collected in the `flatsource` dirctory to your target system. You may find this command:

   ```
   tar chf ~/deMon-source.tar *
   ```

   useful for following symbolic links.

5. Build deMon on your target system, using the minimum necessary force.

If you experience errors while running deMon shell scripts, check your operating system documentantion for instructions on enabling POSIX conformance. Some hints for the `Tru64` and `Solaris` operating systems are found in `$deMon/doc/os/`.
If you experience mysterious crashes running compiled binary, try some of the following:

- Try increasing process limits, such as the stack size limit, data segment size limit, or memory size limit. If you use Bourne shell or derivative (sh, ksh, bash, ...), the appropriate command is `ulimit`. For C-shell or its ilk (csh, tcsh, ...), the right command is `limit`. On some operating systems tuning kernel parameters may be required to adjust the limits.

- Try building deMon without using any external libraries – the libraries may be incompatible with the compilation options in the deMon makefile, or may not support functions needed by deMon properly. Debugging and generic versions typically do not require external libraries.

- Try decreasing optimization level – see section 2.2.4 and appendix B for the instructions on modifying platform configuration files.

If your compiled binary runs, but fails some of the tests, you can try some of the following:

- Try decreasing optimization level – see section 2.2.4 and appendix B for the instructions on modifying platform configuration files. The following routines are particularly numerically sensitive, and may be affected by aggessive compiler optimizations:

tensor.f	Calculation of the molecular inertia tensor
rs.f	EISPACK diagonalization routine
jacobi.f	Jacobi diagonalization routine
pythag.f	auxiliary routine for jacobi.f
dsyev.f	LAPACK diagonalization routine
dsyevd.f	LAPACK diagonalization routine (divide and conquer)
dsyevr.f	LAPACK diagonalization routine (relatively robust representations)
dgelss.f	LAPACK least-squares fit routine
dpptrf.f	LAPACK routine for Cholesky factorization
dpptri.f	LAPACK routine for matrix inversion using Cholesky factorization
dtptri.f	LAPACK routine for triangular matrix inversion
blas1.f	BLAS Level 1 routines
blas2.f	BLAS Level 2 routines
dgemm.f	BLAS Level 3 DGEMM routine
dsyrk.f	BLAS Level 3 symmetric rank k update

- Try enforcing strict conformance to the IEEE-754 floating point standard. Your compiler documentation should provide the instructions for enforcing the conformance.

- If you are building deMon with an external BLAS or LAPACK libraries, try building using Fortran routines supplied with deMon itself. Sometimes, vendor-supplied libraries are trading off accuracy for extra speed.

- If you are building deMon using Fortran BLAS or LAPACK, included with the deMon distribution, try building with vendor-supplied numerical libraries. Sometimes, vendor compilers have issues with generating numerically accurate code for these routines, but hand-optimized numerical libraries work around these issues.

If none of these suggestions help, try searching the archives of the deMon users mailing list, and deMon developers mailing list. Both archives are found at `http://www.deMon-software.com/`. If you cannot find the solution to your problem in the archives, you can try sending your questions to one of the mailing lists. The addresses of the lists can be found on the same web site.

Finally, if everything else fails, you can ask deMon gatekeeper for assistance. The gatekeeper's e-mail is: `deMonkey@deMon-software.com`. **Important:** Please do not bother telling to the gatekeeper, or to any of the developers: *"Your stuff does not work, and it is just a piece of junk"*. This kind of approach will only pi** off people who put a lot of effort into making sure that everything works as smoothly as it possibly could.
Instead, please provide answers to these questions:

- Which version of deMon are you using? (Check `$deMon/source/deMon.f`).
- Which hardware platform are you using? The processor and system model are important.
- Which operating system are you using? (Try `uname -a` shell command).
- Which compiler are you using? Please give both the name and the version.
- Which numerical libraries (if any) are you using? Both name and version are important.
- Which options for the build platforms are you offered by the `./install.sh` script?
- Which build platform did do you choose?
- If your build fails while running `./install.sh`, what are the contents of the the `build-*.log` file it produced in the `makefiles` subdirectory?
- If you experience failures in the test set, which test cases fail?
- If you experience failures in a specific test case, please include *both* the `*.out` and `*.err` files from the corresponding subdirectory of `examples` directory.

While doing all this, please keep in mind that people you are asking for assistance are just as busy as you are. Please try to help them to resolve *your* problem.

2.6 How to Run deMon

The use of deMon requires the preparation of the input file (see Chapter 4), job execution and interpretation of the output (see Chapter 5). To run the program, three files are necessary:

`deMon.inp` is the input file.
`AUXIS` contains the Gaussian auxiliary function.
`BASIS` contains the Gaussian orbital basis functions.

`AUXIS` and `BASIS` are ASCII files which are provided with the program. It is possible to run deMon executable binaries, found in `$deMon/bin/*.x`, directly. In this case the above three files have to be present in the current directory. The job can be executed by the command `$deMon/bin/*.x &` or, without hangups, by `nohup $deMon/bin/*.x &` in the background. The current directory is used as working directory and all necessary scratch files are created here. A second possibility, less cumbersome, is to use the standard deMon driver script, provided in `$deMon/bin/deMon`. You may wish to add the `$deMon/bin` directory to your command search path, to avoid repeatedly typing the complete path to the script. The deMon driver script is invoked as follows:

```
deMon [option ...] job_name
```

The driver script will attempt to determine the location of the deMon installation directory, and will supply reasonable defaults for most options. Additionally, the following options can be specified:

-a	auxis	Specify alternate basis set file. The default fiting basis set file is looked up in the current directory, then in deMon installation directory.
-b	basis	Specify alternate basis set file. The default is looked up exactly as for the auxis.
-d	debugger	Specify an alternative debugger (the default is gdb)
-e	ecp	Specify alternate ECP file. The default is looked up as described for auxis.
-g		Start deMon interactively, in a debugger
-p		Print the name of deMon binary to be used, and exit
-h		Print summary of the available options
-r	directory	Use “directory” to store restart files, and other large files, produced by the simulation. By default, it is the current directory
-s	directory	Use “directory” for scratch files
-v		Be verbose
-q		Be quiet

-x program Use an alternative deMon binary

The deMon driver expects to find the job input (using the syntax, described in Section 3.3) in the file specified on the command line, with the extension `.inp` attached. Depending on the job type, additional data have to be provided in files with the same base name, and extensions `.cub`, `.lat`, `.rst`, and `.qmd`.
The main deMon output is stored in a file with the extension `.out`. Additionally, some system information is provided in a file with the same base name, and extension `.err`. Depending on the nature of the job, additional output files, with extensions `.asc`, `.bin`, `.lat`, `.mkl`, `.mol`, `.new`, `.pie`, `.rst`, `.nmr11`, `.nmr70`, `.nmr71` and `.nmr72` may be generated. The driver script takes care not to delete any previously existing output files. If any of the output files with these extensions exist at the time execution begins, the driver will create backup copies. These copies will have an additional extension `.bak`, appended to their full name (if any of the files with these extensions exist, they will be removed).
Note that most of the deMon output files are originally created in the scratch area, and are only copied to the job starting directory after the job completes. The only exception is the system information file (with the `.err` extension). If you need to examine the output of a running deMon job, you will find the exact location of the scratch directory in the system information file.

3 Carrying On

3.1 Parameters

Table 2 summarizes the limits of the shipped deMon version for memory and disk space requirements, number of atoms as well as basis and auxiliary functions. These limits are specified in the section `user defined parameters` of the `parameter.h` file. (This file is found in the directory `$deMon/include`.) If a calculation violates these limits the program stops and prints PARAMETER EXCEEDED statement(s) for the parameters that have to be changed. In this case, please change only the first exceeded parameter (the following ones may be corrupted by array bound violations) to the suggested (USED) value and recompile the program. Repeat this procedure until all PARAMETER EXCEEDED statements have disappeared. It is recommended not to increase the MAXRAM value over the physical memory (RAM) available, because otherwise it may result in large paging overheads during program execution.
The disk requirements are dominated by the size of the electron repulsion integral scratch file `ioeri.scr`, in the case of a conventional SCF calculation, and by the files `ioscf.scr`, `iocdf.scr` and `iogrd.scr` in the case of direct SCF calculations (see 4.4.1).

3.2 Files

All files in deMon are opened with explicit filenames, rather than using operating system-specific compiler defaults. Throughout this guide, the name used in the Fortran OPEN statement is referred to as the logical filename. These logical filenames are read by the program or created in the working directory.
The logical files are connected to Fortran I/0 units in the file `fileio.h`. Table 3 contains the logical and physical filenames used in the program.
Besides these files, special plot files may be created for the visualization of molecular fields, like the electronic density, electrostatic potential, etc. with VU (see Chapter 8). These files are named according to the molecular field to be visualized. For example, a file `RHO.bin` is created for the visualization of the electronic density. The `deMon.pie` file is the VU control file and refers in this case to the `RHO.bin` file. If the job is executed by a script, care has to be taken that these files are copied from the working directory to the output path and that the filenames are consistent with the ones in the `deMon.pie` file. These entries can also be modified manually in order to use the same VU control file

Table 2: Parameter settings in the shipped deMon version.

Parameter	Setting	Description
MAXDISK	32768	Maximum disk size for scratch files in Mbytes
MAXRAM	8192	Maximum RAM size for program kernel in Mbytes
MAXATOM	5000	Maximum number of atoms
MAXAUX	25000	Maximum number of auxiliary functions
MAXAUXSET	10000	Maximum number of auxiliary function sets
MAXAUXSHL	10000	Maximum number of auxiliary function shells
MAXCON	21	Maximum degree of contraction
MAXCUBE	100000	Maximum number of cube (embedding) points
MAXGTO	25000	Maximum number of primitive Gaussian functions
MAXECPGTO	25000	Maximum number of ECP Gaussian functions
MAXECPSHL	5000	Maximum number of ECP shells
MAXLAUX	6	Maximum L quantum number for auxiliary functions
MAXLBAS	5	Maximum L quantum number for basis functions
MAXLECP	5	Maximum L quantum number for ECP functions
MAXSHL	10000	Maximum number of orbital basis shells
MAXSTO	15000	Maximum number of contracted (STO) orbitals
MAXTBSTO	25000	Maximum number of tight-binding (STO) orbitals
MAXADJ	25	Maximum number of close, bond-line contacts per atom

for different molecular fields.

3.3 Input Syntax

The input of deMon is easy and mnemonic and at the same time offers high flexibility. The input file contains keywords, options and keyword bodies. A keyword forms together with its options and the keyword body a keyword block. The ordering of these blocks is free. All keywords, with exception of GEOMETRY, have associated default values, which are used, if the keyword is not explicitly specified in the input. Table 4 summarizes the special symbols allowed in the job input file `deMon.inp`.

The input lines in the job input file `deMon.inp` are restricted to 160 characters. The input is not case sensitive. A keyword line can be continued by the `&` symbol at the very end of the line. At most, five continuation lines are permitted. **The TITLE line cannot be continued and the length of the title string is restricted to 60 characters**.

Table 3: Logical and physical filenames in deMon. The I/0 units are given in parentheses.

Logical Filename	Physical Filename		Description	Type
AUXIS	AUX	(1)	Auxiliary function file	ASCII
BASIS	BAS	(2)	Basis set file	ASCII
ECPS	ECP	(3)	ECP file	ASCII
deMon.cub	CUB	(4)	Embedding file	ASCII
deMon.inp	INP	(5)	Input file	ASCII
deMon.out	OUT	(6)	Output file	ASCII
deMon.new	NEW	(7)	New input file for restart	ASCII
deMon.mol[a]	MOL	(8)	MOLDEN input file	ASCII
deMon.mkl[a]	MOL	(8)	MOLEKEL input file	ASCII
deMon.pie	PIE	(9)	VU control file	ASCII
deMon.rst	RST	(10)	RESTART file	BINARY
deMon.lat	LAT	(11)	Plot lattice file	ASCII / BINARY
deMon.qmd	MDRST	(12)	MD restart file	ASCII
deMon.nmr11	NMR11	(13)	NMR grid file	BINARY
deMon.nmr70	NMR70	(14)	NMR dimension file	BINARY
deMon.nmr71	NMR71	(15)	NMR pointer file	BINARY
deMon.nmr72	NMR72	(16)	NMR orbital file	BINARY
deMon.mag	MAGTAPE	(17)	Alternative NMR file	BINARY

[a]The I/O unit 8 is connected either to the MOLDEN or MOLEKEL input file depending on the VISUALIZATION (4.9.1) keyword. For the XYZ output the MOLDEN file (deMon.mol) is used in any case.

Table 4: Special symbols of the deMon input.

Symbol	Description
&	Continuation of a keyword line
#	First character of a comment line
=	Assignment of numerical values to options
-	Combination of keyword options
Space	Separation of keyword and options
,	Separation of different options (optional)
END	End of a keyword data block

The symbol **#** indicates a comment line. Comment lines can only occur between complete keyword data blocks including the keyword body, if existing. Comments inside a keyword

line or body result in unpredictable read operations. The end of a keyword block can be indicated by an `END` statement or comment symbol `#` in a new input line or simply by a new keyword line.
The keyword options have to be given directly after the keyword, separated by space(s). If they are not present the default option setting for the keyword is used. Numerical values are assigned by the `=` sign to keyword options (e.g. `SCFTYPE Tol = <Real>`, where `<Real>` indicates a real number). For the combination of keyword options the `-` sign is reserved (e.g. `VXCTYPE B88-LYP`). The keyword body starts in a new input line directly after the keyword line(s), and must have the appropriate structure given in Chapter 4, *"Keywords"*.

4 Keywords

The description of the Keywords is grouped according to their functionality into geometry input, basis set input, electronic state control, self-consistent field (SCF) control, optimization control, frequency control, property control, environmental control and interface control keywords. Keywords which don't fit into these groups are listed at the end of this chapter under miscellaneous keywords. Keyword options which exclude each other are listed in one line separated by / signs. If these options are not specified in the input the underlined one is used by default. If more than one of these options are given the last one will override the previous ones. The boldface printed part (first five letters) of the keywords and options are mandatory for the input. Therefore, **OPTIM**IZATION, **OPTIM**IZE and **OPTIM** are all allowed input forms for the keyword OPTIMIZATION.

4.1 Geometry Input

4.1.1 Keyword **GEOME**TRY

This keyword is mandatory! It specifies the molecular geometry.
Options:
CARTESIAN / **ZMATR**IX / **MIXED**

CARTESIAN	The molecular structure is given in Cartesian coordinates.
ZMATRIX	The molecular structure is defined by a Z-Matrix.
MIXED	The first atoms of the molecular structure are defined by Cartesian coordinates and the following ones are defined by a Z-Matrix.

ANGSTROM / **BOHR**

ANGSTROM	Coordinates or bond distances are given in Ångström.
BOHR	Coordinates or bond distances are given in atomic units.

Description:
The geometry is read in the keyword body of GEOMETRY in free format, one line for each atom. In the case of a CARTESIAN input the atomic symbol (e.g. H), which may carry an identification number (e.g. H1), and the x, y and z coordinates of each atom of the system have to be specified. As an example the geometry of H_2O may be specified as:

```
GEOMETRY CARTESIAN ANGSTROM
O  0.00  0.00  0.00
H  0.76  0.00  0.52
H -0.76  0.00  0.52
```

The Cartesian coordinates of an individual atom, defined by the atomic symbol (e.g. H2), or an atom group, defined by the element symbol (e.g. H), can be frozen during the geometry optimization. In order to keep all three degrees of freedom of an atom fixed the corresponding atomic symbol or element symbol has to be specified with the string `XYZ` in the keyword body of CONSTANTS. The string `X` will only freeze the x coordinate of the atom, the string `XY` the x and y coordinates, and so on. In the following example the oxygen atom is kept frozen and the movements of all hydrogen atoms are restricted to the xz-plane (y coordinate frozen) during the optimization.

```
GEOMETRY CARTESIAN ANGSTROM
O  0.00  0.00  0.00
H  0.76  0.00  0.52
H -0.76  0.00  0.52
#
CONSTANTS
O XYZ
H Y
```

As usual, the comment line (`#`) is only given for clarity. The effective nuclear charge (in atomic units), which is identical to the number of electrons counted for the atom, and the nuclear mass (in atomic mass units; amu) can be specified by an integer and real number after the coordinates, respectively. Therefore, the following input for water consists of a frozen oxygen atom, a hydrogen atom, `H1`, fixed in the xz-plane with the effective nuclear charge 0 (no electrons are counted for this hydrogen) and another hydrogen atom, `H2`, with the effective nuclear charge 2 and the nuclear mass 2.014 (Deuterium).

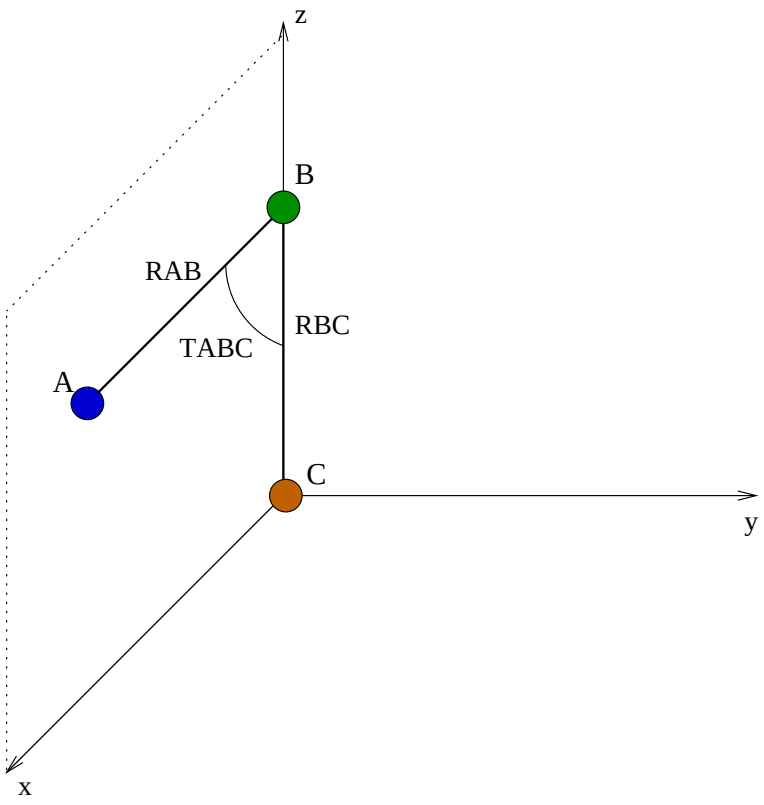
```
GEOMETRY CARTESIAN ANGSTROM
O   0.00  0.00  0.00
H1  0.76  0.00  0.52  0
H2 -0.76  0.00  0.52  2  2.014
#
CONSTANTS
O  XYZ
H1 Y
```

If quantities are not specified the default values are used (e.g. the `O` atom in the above example has a nuclear charge from 8 and a mass of 15.999 amu).

If the geometry of the system is given by a Z-Matrix, using the option ZMATRIX, each line in the keyword body of GEOMETRY describes the connectivity of the atom, which

again is defined over its atomic symbol. The first atom (C) of the Z-Matrix defines the origin of the input (see Figure 1). No connectivity information should be given for this atom.

Figure 1: Orientation of the first three atoms defined in the Z-Matrix.



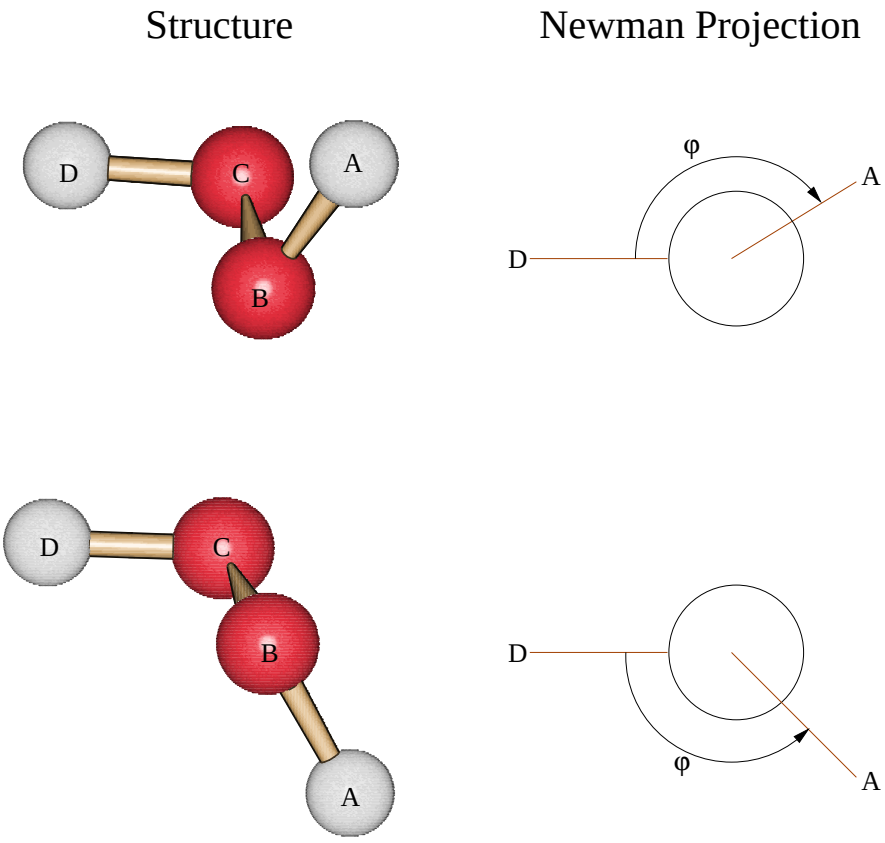
The second atom (B) lies on the z-axis of the input coordinate system and is found at the distance `RBC` from atom C. The third atom (A) lies in the xz-plane, at a distance `RAB` from atom B. The two atoms A and B form angle `TABC` with the first atom (C).
All the remaining atoms are defined by input line(s) in the free format, containing the information:

LABEL NB RAB NC TABC ND PABCD

Here LABEL is the atomic label of atom A. NB, NC and ND are the Z-Matrix line numbers or the atomic labels of the atoms B, C and D over which atom A is defined by a length (RAB), angle (TABC) and dihedral angle (PABCD). These are the so-called internal coordinates of the system. There values are given by RAB (in Ångström or Bohr), TABC and PABCD (in degrees), respectively. The dihedral angle is defined by the angle between the planes spanned by the atoms A, B, C and B, C, D, respectively. The sign of

the dihedral angle is defined according to the Newman projection shown in Figure 2. If the projection angle is oriented clockwise, the dihedral angle is taken to be positive.

Figure 2: Definition of positive (top) and negative (bottom) dihedral angles in deMon.



Instead of giving the values for RAB, TABC and PABCD directly in the Z-Matrix they may also be represented by symbolic strings, up to eight characters long. These labels have to be defined in the CONSTANT and VARIABLE input sections (see 4.1.2 and 4.1.3). As an example, ethene C_2H_4 can be defined as:

```
GEOMETRY ZMATRIX ANGSTROM
C1
C2  C1  rCC
H1  C1  rCH  C2  aCCH
H2  C1  rCH  C2  aCCH  H1  dHCCH
H3  C2  rCH  C1  aCCH  H2  dHCCH
H4  C2  rCH  C1  aCCH  H1  dHCCH
#
VARIABLES
rCC   1.3390
```

```
rCH   1.0850
aCCH  121.09
#
CONSTANT
dHCCH 180.0
```

The internal coordinates listed under VARIABLES may change during geometry optimization, whereas the coordinates given under CONSTANTS are kept frozen.
If one uses internal coordinates, dummy atoms are often useful. The atomic symbol X has been reserved for them. The following example shows the use of a dummy atom in the definition of HCN.

```
GEOMETRY ZMATRIX ANGSTROM
H
C  1  R1
X  2  1.0  1  90.0
N  2  R2   3  A1    1  180.0
#
VARIABLES
R1  1.089
R3  1.166
A1  90.00
```

Here the dummy atom is used to avoid a 180° angle which may cause problems in a geometry optimization. Example 5.1 shows the use of a dummy atom for the definition of a ring system. In order to facilitate the input of the internal coordinates for some special cases, alternative geometrical parameters can be given instead of angles and dihedral angles in the Z-Matrix input. The substitutions are indicated by string codes, which may appear after the connectivity information in each Z-Matrix line. The possible substitution codes are listed in Table 5.

Table 5: Substitution codes for the internal coordinate substitution in the Z-Matrix.

Subst. Code	Internal Coordinates			Description
RAD	RAB	TABC	PABCD	Default: distance-angle-dihedral
RRD	RAB	RAC	PABCD	Angle ABC is substituted by length AC
RAA	RAB	TABC	TABD	Torsion ABCD is substituted by ABD angle
RRA	RAB	RAC	TABD	Combination of RRD and RAA
RRR	RAB	RAC	RAD	Atom specified by three distances

Similar to the CARTESIAN input, the atomic mass and effective nuclear charge of an atom can be optionally specified at the end of each Z-Matrix line. The examples 5.2 and

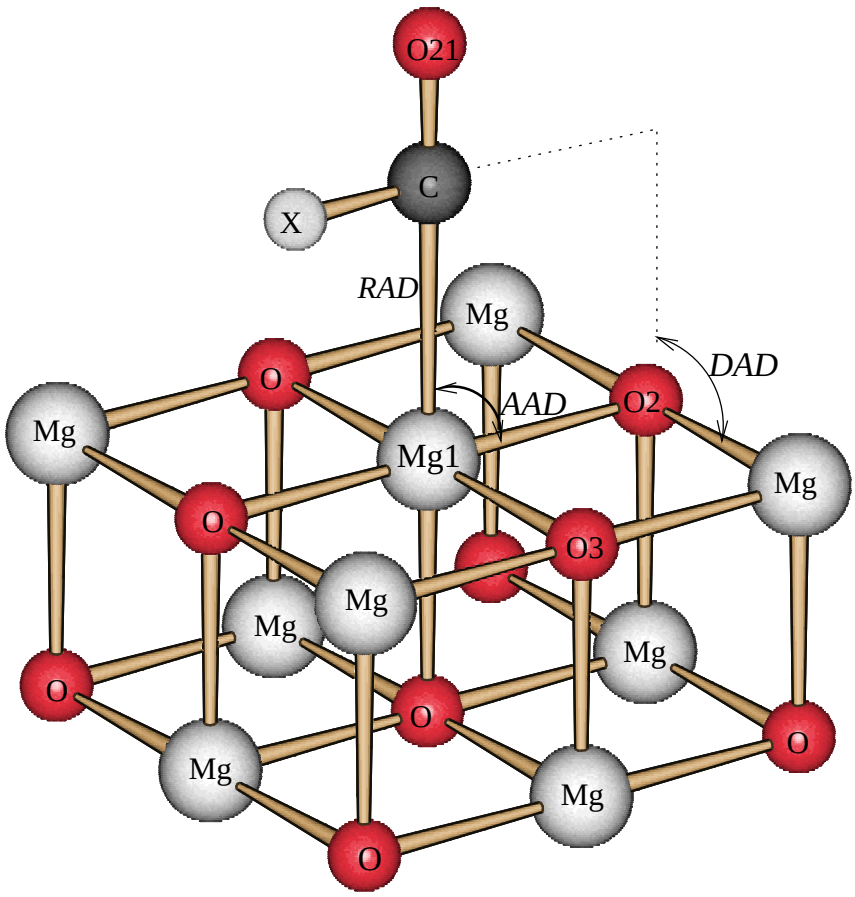
5.3 show some common applications of Z-Matrix inputs.
If the geometry of the system is given in the MIXED format, the first atoms are defined using their Cartesian coordinates. At least the first three atoms (some which may be dummy atoms), have to be defined in this way. The following atoms can then be specified in the Z-Matrix format, as described above. The reference atoms can be defined using either Cartesian or internal coordinates. The coordinate system is defined by the first three atoms. The following example (see Figure 3), shows a MIXED input of a CO molecule adsorbed on a MgO cluster:

```
GEOMETRY MIXED ANGSTROM
Mg1     0.000    0.000    0.000
O2      0.000    2.105    0.000
O3      2.105    0.000    0.000
O4      0.000   -2.105    0.000
O5     -2.105    0.000    0.000
Mg6     2.105    2.105    0.000
Mg7     2.105   -2.105    0.000
Mg8    -2.105   -2.105    0.000
Mg9    -2.105    2.105    0.000
O10     0.000    0.000   -2.105
Mg11    0.000    2.105   -2.105
Mg12    2.105    0.000   -2.105
Mg13    0.000   -2.105   -2.105
Mg14   -2.105    0.000   -2.105
O15     2.105    2.105   -2.105
O16     2.105   -2.105   -2.105
O17    -2.105   -2.105   -2.105
O18    -2.105    2.105   -2.105
C19     Mg1      RAD      O2       AAD      O3       DAD
X20     C19      1.0      Mg1      90.0     O2       180.0
O21     C19      RCO      X20      90.0     Mg1      180.0
#
CONSTANTS
Mg    XYZ
O2    XYZ
O3    XYZ
O4    XYZ
O5    XYZ
O10   XYZ
O15   XYZ
O16   XYZ
O17   XYZ
O18   XYZ
#
VARIABLES
```

```
RAD  2.0
AAD  90.0
DAD  90.0
RCO  1.4
```

In the example above, geometry of the MgO cluster is fixed by freezing its Cartesian coordinates (see 4.1.2). The carbon atom of the CO molecule is defined with respect to the `Mg1`, `O2` and `O3` plane of the Cluster (see Figure 3) by the internal coordinates `RAD`, `AAD` and `DAD`. These coordinates optimized because they are declared as variables (see 4.1.3). The same holds for the CO bond length that is defined by the internal coordinate `RCO`. The dummy atom `X20` has to be used in order to avoid an ill-defined dihedral angle if the `O21`, `C`, `Mg1` arrangement becomes linear. Therefore, the above example describes an input for the free optimization of an CO molecule, including internal relaxation, on a frozen MgO cluster surface.

Figure 3: Coordinate definition of the above MIXED input.



For a hybrid QM/MM calculation, additional parameters can be added *at the end* of each geometry specification line. These parameters are in the keyword form (parameter=value), and do not need to appear in any particular order. QM/MM keywords are recognized, and behave identically in all three input formats. The QM/MM-related geometry keywords are:

Keyword	Value	Description
QMMM=		Specifies handling of this atom when QM/MM is activated:
	MM	This atom is a part of the molecular-mechanical subsystem
	QM	This atom is a part of the quantum-mechanical subsystem
	LINK	This a bond-terminating atom, at the QM/MM boundary. The default is 'MM' if QM/MM is activated, and 'QM' if it is not.
CAP=		Specifies choice of the capping potential for a link atom:
	ZLY1	"3-21G" pseudohalogene capping potential of Ref. [10]
	ZLY2	"6-31G*" pseudohalogene capping potential of Ref. [10]
	DHC1	"QCP1" pseudohydrogen capping potential of Ref. [11]
	DHC2	"QCP2" pseudohydrogen capping potential of Ref. [11]
	DHC3	"QCP3" pseudohydrogen capping potential of Ref. [11]
	DHC4	"QCP4" pseudohydrogen capping potential of Ref. [11]
	(*ecp*)	Use specified potential from the ECP library
MMTYPE=	*type*	Specifies forcefield atom type. Atom types are forcefield-specific. The list of the known atom types for the built-in UFF force field is given in section 6. MMTYPE is relevant for all types of atoms, including QM and LINK atoms.
	GUESS	A special atom type GUESS will instruct deMon to try to guess the appropriate force field type (see below). This is the default.

`Q=` *charge* This option can be used to specify partical charge on the MM atom, in proton charge units. It is up to the forcefield to use, or not to use this charge.

`GUESS` A special charge `GUESS` (which is the defalt) will instruct deMon to guess the appropriate value.

`BOND=` *list* Specifies the list of explicit bonds for this atom. Bond specification contains a list of colon-separated atom ranges, each followed by optional slash sign and the bond order. Both atom numbers (in the input order) and atom labels are recognized; atom labels must refer to a previously defined atom. There are no restrictions on atom numbers.

A few examples:

`3` Bond to atom 3; the bond order will be guessed automatically.

`C_PHEN/1.5` Bond of the order 1.5 to a previously defined atom with the label `C_PHEN`

`15-17` Bonds to atoms 15 through 17, with automatically defined orders

`1/2:2-3/1.0` Double bond to atom 1; single bonds to atoms 2 and 3

It is OK to specify a bond for only one of the connected atoms. If `FORCEFIELD BONDS=AUTO` is in effect, explicit bonds take precedence over the guessed bonds.

`NOBOND=` *list* Specifies that no bond should be formed to these atoms. The syntax of the atom ranges is identical to the `BOND=` option. `NOBOND=` overrides automatically guessed bonds.

Keywords QMMM=, CAP=, MMTYPE=, and Q= can only appear once for each atom. Multiple instances of the BOND= and NOBOND= keywords are allowed. Their effect is cumulative. QM/MM keywords are recognized, but ignored by deMon in a QM-only calculation (QMMM QM, see section 4.5.1).

An example of a QM/MM geometry input is:

```
GEOMETRY Z-MATRIX
C1                                    QMMM=QM
O1     1 RCO                          QMMM=QM Q=-1.0 BOND=C1/1.0
H1     1 RH1     2 AH1                QMMM=QM
H2     1 RH2     2 AH2    3 DH2       QMMM=QM
O_W1   1 RCW1    2 ACW1   3 DCW1      Q=-0.834
H3     1 RH3     2 AH3    3 DH3       QMMM=QM
H_W11  5 RH1W1   1 AH1W1  2 DH1W1     Q=+0.417 BOND=O1/0.03:O_W1/0.90
H_W12  5 RH2W1   7 AH2W1  1 DH2W1     Q=+0.417
O_W2   1 RCW2    2 ACW2   3 DCW2      QMMM=QM Q=-0.834
H_W21  9 RH1W2   1 AH1W2  2 DH1W2     QMMM=QM Q=+0.417 &
                                      BOND=O1/0.03:O_W2/0.90
H_W22  9 RH2W2  10 AH2W2  1 DH2W2     QMMM=QM Q=+0.417
#
VARIABLES
RCO           1.37257
RH1           1.11599
 ...
AH2W2       104.0
DH2W2         0.0
```

The full input for this caluclation is discussed in section 5.4.

4.1.2 Keyword CONSTANTS

This keyword assigns constant Cartesian and internal variables.

Options: None

Description:

In the body of CONSTANTS all atoms and internal coordinates which are kept frozen during the geometry optimization are listed. The atomic or element symbol with the string XYZ is used to freeze the atom or group. The string X will only freeze the x coordinate of the atom, the string XY the x and y coordinates, and so on.

For internal coordinate inputs the symbolic coordinate string of the Z-Matrix and its value (real number) is given in order to freeze this degree of freedom. In the case of a

MIXED input this two syntax forms can be mixed (see Figure 3 and the corresponding input example).

```
Constants
H    XYZ
A  120.0
```

Here the first line freezes all hydrogen atoms to their positions. In the second line the internal coordinate `A` is set to 120° and kept frozen. Note that all parameters listed in `CONSTANTS` section are treated as `VARIABLE`s in a molecular dynamics simulation.

4.1.3 Keyword **VARIA**BLES

This keyword assigns variable internal coordinates.
Options: None
Description:
The body of VARIABLES collects all symbolic coordinate strings of the Z-Matrix with their values (real number) which should be optimized (see 4.1.1 for examples).

4.1.4 Keyword **SYMME**TRY

This keyword controls the symmetry analysis of the molecular structure.
Options:
OFF / **ON**

OFF	No symmetry analysis is performed.
ON	A symmetry analysis is performed.
DETECT=ON	Symmetry detection is performed. This is the default
DETECT=OFF	No symmetry detection is performed.

Description:
If a symmetry analysis is performed the point group of the molecule is automatically detected and the molecular INPUT ORIENTATION is transformed into the STANDARD ORIENTATION defined by the center of mass and the main axes of the tensor of inertia. All further calculations are done in this orientation. If no symmetry analysis is performed the INPUT ORIENTATION is used for the SCF calculation. However, if a geometry optimization or a frequency analysis is requested the STANDARD ORIENTATION is

used in any case (see 5.1). For atoms, the use of the SYMMETRY keyword enforces the zeroing of the off-diagonal elements of the Kohn-Sham matrix for different angular momenta.

Currently, deMon symmetry analyser supports C_n, C_{nv}, C_{nh}, D_n, D_{nd}, and D_{nh} point groups with $n \leq 6$. The rotation-inversion groups S_n are supported for $n \leq 8$. If a higher group is present in a molecule, deMon will abort the calculation, even when `SYMMETRY OFF` is specified. Symmetry analysis can be suppressed altogether by including `DETECT=OFF` on the symmetry line (See example 5.5.

If you are uncertain about a symmetry assignment, made by deMon, or are wondering whether your system is *approximately* symmetric, you may find an unsupported program `$deMon/unsupported/symmetry.c` useful.

4.2 Basis Set Input

4.2.1 Keyword BASIS

This keyword specifies the basis set.
Options:

<Basis>	The basis set string <Basis> defines the global basis set. If absent, the DZVP basis set is used by default.

Description:
In the BASIS keyword body basis sets can be assigned to individual atoms according to the atomic symbol (e.g. H4) or to atom groups over based on the element symbol (e.g. H). The global basis set is used for all atoms which are not defined by either of these mechanisms. Specifying QM/MM capping potential will override basis set selection, made with the `BASIS` keyword (see section 6). More specific definition of the basis set overrides a less specific one, so that:
QM/MM `CAP=` overrides:
<atomic symbol> overrides:
<element symbol> overrides:
<global basis set>
Thus any basis set definition for an atom can be overriden by the explicit assignment of the basis set using the atomic symbol. E.g. for C_2H_4 (see input in 4.1.1) the following basis set definition:

```
Basis (DZVP)
C     (TZVP)
C1    (STO-3G)
```

assigns a STO-3G basis to atom C1, a TZVP to C2 and the DZVP basis to all other atoms. Instead of using basis set strings, the basis set of an atom may also be specified by the Huzinaga notation given in the `BASIS` file. Using this notation the above basis set definition would read as:

```
Basis (DZVP)
C     (7111/411/1*)
C1    (33/3)
```

The basis set file `BASIS` contains the basis sets listed in Table 6. Other basis sets can be obtained from the Extensible Computational Chemistry Environment Basis Set Database [12] at *http://www.emsl.pnl.gov/forms/basisform.html* by choosing the deMon basis set

format. Instead of reading the basis set from the BASIS file the user can also define it directly in the input file following the format:

```
Basis
SYMBOL Read
N  L  K
   EXPONENT  COEFFICIENT
      :          :
   EXPONENT  COEFFICIENT
```

where `SYMBOL` can be an element or atomic symbol, `N` and `L` are the main and angular momentum quantum number of this shell and `K` is the contraction degree. Here, a shell collects all contracted orbitals of the same angular momentum quantum number like p_x, p_y, p_z or d_{xx}, d_{xy}, d_{xz}, d_{yy}, d_{yz} and d_{zz}. The contracted orbitals, $\mu(\mathbf{r})$, are linear combination of (atom centered, normalized) Gaussian type orbitals (LCGTO–Ansatz), which are called the primitive orbitals, $g(\mathbf{r})$:

$$\mu(\mathbf{r}) = \sum_{k=1}^{K} d_{\mu k}\, g_k(\mathbf{r}) \tag{4.1}$$

$$g_k(\mathbf{r}) = (x - A_x)^{a_x}\,(y - A_y)^{a_y}\,(z - A_z)^{a_z}\, e^{-\zeta_k(\mathbf{r}-\mathbf{A})^2} \tag{4.2}$$

The exponents ζ_k and contraction coefficients $d_{\mu k}$ are listed in free format under the shell definition line, one line for each primitive orbital (EXPONENT and COEFFICIENT). A user defined basis set input is shown in Example 5.6.

4.2.2 Keyword AUXIS

This keyword specifies the auxiliary function set.
Options:

<Auxis>	The auxiliary function set string <Auxis> defines the global auxiliary function set. If absent, the A2 auxiliary function set is used by default.

Description:
Similar to the basis sets the auxiliary function sets can also be assigned to individual atoms according to the the atomic symbol or to atom groups based on the element symbol in the keyword block of AUXIS. The syntax and hierarchy of these assignments are similar to the basis set assignments (see 4.2.1). For C_2H_4 (see input 4.1.1) the following auxiliary function definition:

Table 6: Basis sets available in the deMon `BASIS` file.

Basis Set	Elements	Description
DZV	H, C	Double ζ basis set for saturation only
DZVP	H–Xe	LDA double ζ polarization[a] basis set [13]
DZVP2	Be–F, Al–Ar, Sc–Zn	Modified DZVP basis set
TZVP	H, Li, C–F, Si–Cl	LDA optimized triple ζ polarization basis set
TZVP-FIP1	H, C–F, S, Cl	TZVP with field-induced polarization [15,16]
TZVP-FIP2	H, C–F, S, Cl	for α, β (FIP1) and γ (FIP2) calculations
SADLEJ-FIP	H, C, N, O, F	
EPR-III	H–F	EPR basis set [17]
IGLO-II	H–F, Si, Cr, Fe	NMR basis set [18]
IGLO-III	H–F, Si, Cr, Fe	NMR basis set [18]
STO-3G	H–I	STO–3G basis set for testing only [19]
SAD	H, C–F	Sadlej FIP basis set [20]
LIC	H–Ne	Lie-Clementi basis set [21]
WACHTERS	Sc–Cu	Wachters basis set without f functions [22]
DZ-ANO	H–Zn	Double ζ ANO basis set from Roos [23]
Ahlrichs	H–Kr	Ahlrichs basis sets: A-VDZ, A-PVDZ, A-VTZ, A-PVTZ
ECP-SD		
EECP-SD		
QECP-SD		

[a]Modified for Li and Na [14].

```
AUXIS (A2)
C     (GEN-A2*)
C1    (GEN-A2)
```

assigns a GEN-A2 auxiliary function set to atom C1, a GEN-A2* to atom C2 and the A2 auxiliary function set to all other atoms. The `AUXIS` file of deMon contains only the A2 auxiliary functions set. The other auxiliary function sets used above (`GEN-A2` and `GEN-A2*`) are automatically generated according to the procedure described in Appendix A. Additionally, the auxiliary function sets can be directly specified in the input file using

the format:

```
Auxis
SYMBOL Read
LMAX  EXPONENT
 :       :
LMAX  EXPONENT
```

where `LMAX` denotes the maximum angular momentum quantum number of the auxiliary function set and `EXPONENT` the exponent which is shared [24,25] by all functions in the set. Thus an auxiliary function set with `LMAX` = 2 contains ten functions, namely one s, three p and six (Cartesian) d functions. In deMon these functions are primitive Hermite Gaussians of the form (without normalization):

$$\bar{a}(\mathbf{r}) = \left(\frac{\partial}{\partial A_x}\right)^{\bar{a}_x} \left(\frac{\partial}{\partial A_y}\right)^{\bar{a}_y} \left(\frac{\partial}{\partial A_z}\right)^{\bar{a}_z} e^{-\zeta(\mathbf{r}-\mathbf{A})^2} \tag{4.3}$$

The input of an user defined auxiliary function set is described in Example 5.6.

4.2.3 Keyword ECPS

This keyword specifies the effective core potentials (ECPS).
Options:

<ECP>	The ECP string <ECP> defines the global effective core potentials. If absent, an all-electron calculation is assumed by default.

The keyword ECPS is very similar to the keyword BASIS (Section 4.2.1). Different ECPs can be assigned to individual atoms according to the atomic symbol (e.g. Au1) or to atom groups based on the element symbol (e.g. Au). The global ECP is used for all atoms which are not explicitly defined. Requesting a QM/MM capping potential (see section 6 overrides ECP section in the `ECPS` keyword. If ECPS is not specified for an atom but the substring "ECP" is present in the basis set definition, then an ECP with the same name will be automatically assigned to the atom. The assignment of the ECP based on atomic symbols, element symbols and global ECPS definition possesses the hierarchy:

QM/MM `CAP=` overrides:
<atomic symbol> overrides:
<element symbol> overrides:
<global ECPS> overrides:
<basis set name>

Thus any ECPS definition for an atom can be overwritten by the explicit assignment of the ECP using the atomic symbol. E.g. for Au_2O the following ECPS definition:

```
ECPS (ECP|SD)
Au   (ECP19|SD)
Au1  (ECP1|SD)
```

assigns the one-electron ECP denoted by `(ECP1|SD)` to the atom Au1, the 19 electron ECP denoted by `(ECP19|SD)` to the gold atom and the global defined `(ECP|SD)` ECP to the oxygen atom. The file `ECPS` contains the ECPs from Stuttgart [26] in the deMon format. Also the effective core potentials can be directly specified in the input file using the format:

```
ECPS
SYMBOL Read ELECTRONS
N  L  K
   EXPONENT  COEFFICIENT
      :           :
   EXPONENT  COEFFICIENT
```

where `SYMBOL` can be an element or atomic symbol, `ELECTRONS` is an integer number specifying the number of valence electrons, `N` denotes the radial power of the operator, `L` the angular momentum of the effective potential and `K` the contraction degree. The exponents and contraction coefficients are listed in free format under the ECP block definition line, one line for each Gaussian (EXPONENT and COEFFICIENT). A user defined ECP input is shown in Example 5.7.
Effective core potentials in deMon are defined according to the functional form:

$$\begin{aligned} U^{\mathrm{ECP}} &= U_L + \sum_{l=0}^{L-1} \sum_{m=-l}^{l} \left[U_l(r) - U_L(r)\right] |l,m><l,m| \\ U_l(r) &= \sum_{i=1} C_{li} r^n e^{-\zeta_{li} r^2} \end{aligned}$$

The radial power n must satisfy $n \geq -2$. Positive coefficients C_{li} correspond to a repulsive contribution to the potential.

4.2.4 Keyword MCPS

`TO BE WRITTEN BY R. FLORES AND M. KLOBUKOWSKI`

4.3 Electronic State Control

4.3.1 Keyword **MULTI**PLICITY

This keyword specifies the multiplicity of the system.
Options:

<Integer> Multiplicity of the system.

Description:
The default multiplicities are 1 for closed and 2 for open shell systems. The program will check automatically if the defined multiplicity is allowed for a given molecular system and its charge.

4.3.2 Keyword **CHARG**E

This keyword specifies the charge of the system.
Options:

<Integer> Charge of the system.

Description:
The default charge is 0.

4.3.3 Keyword **MOEXC**HANGE

This keyword alters the molecular orbital ordering in the start or restart density.
Options:

<Integer1> Number of spin-α molecular orbital exchanges.
<Integer2> Number of spin-β molecular orbital exchanges.

Description:
The numbers of the molecular orbitals to be exchanged are read in pairs of integers in the keyword body of MOEXCHANGE, one line for each pair. First all spin-α exchanges are read, and then, in the case of an unrestricted calculation (see UKS in 4.4.1), spin-β exchanges are read. The application of MOEXCHANGE is shown in Example 5.8.

4.3.4 Keyword FIXMOS

This keyword fixes a molecular orbital occupations during the SCF by projection.
Options:
FIXED / **ITERA**TIVE

FIXED	The molecular orbital configuration at the beginning of the projection is used as reference configuration.
ITERATIVE	The molecular orbital configuration of the previous SCF cycle is used as reference configuration.
SCF=<Integer>	SCF cycle at which the projection procedure is turned on. By default the projection starts with the first SCF cycle.
TOL=<Real>	SCF convergence tolerance after which the projection procedure is turned on.

Description:
The projection of the molecular orbital configuration is based on the maximum overlap criterion Ω_{aj} in the form:

$$\Omega_{aj} = \sum_{\mu,\nu} c_{\mu a}^{\text{new}} \, S_{\mu\nu} \, c_{\nu j}^{\text{old}} \tag{4.4}$$

where $\{c_{\mu a}^{\text{new}}\}$ are the new molecular orbital coefficients and $\{c_{\nu j}^{\text{old}}\}$ are MO coefficients of the reference configuration. Occupations of a new MOs a is choosen to match the occupation of the most similar MO j of the reference cycle, which occurs when $|\Omega_{aj}|$ is at maximum. Note that `FIXMOS` and `SMEAR` keywords are mutually exclusive. An application of FIXMOS is shown in 5.9.

4.3.5 Keyword SMEAR

This keyword allows fractional orbital occupations close to the Fermi level.
Options:

<Real>	Energy range ΔE [in a.u.] around the Fermi level in which orbitals are fractionally occupied.

Description:
The SMEAR keyword effects only the molecular orbitals within the specified energy interval $[E_{\text{HOMO}} - \Delta E/2\,,\, E_{\text{HOMO}} + \Delta E/2]$. Therefore, the ΔE value should be selected based

on the orbital energy spectrum (see MOS option of PRINT in 4.10.3). The converged fractional orbital occupation is used in any further step of the calculation (optimization, frequencies, properties etc.). `SMEAR` will assign higher occupations to lower-energy orbitals, and will give identical occupations to degenerate MOs. An application of SMEAR is shown in Example 5.10.

4.3.6 Keyword **CONFI**GURE

This keyword controls the atomic configuration and is therefore only applicable for atoms. Options:

MAX=<Integer>	Maximum number of SCF cycles for which the configuration is used. By default it is used for all SCF cycles.
OCCUPY	The (fractional) occupation of the specified atomic configuration will be given explicitly.

Description:
The orbital configuration is defined in the first line of the keyword body of CONFIGURE. In the case of OKS and ROKS calculations (see 4.4.1) two lines, the first for the α-orbital configuration and the second for the β-orbital configuration, are requested. If spherical orbitals (see 4.4.2) are used one integer number for s, three for p, five for d, etc. have to be specified. Empty shells (see 4.2.1 for the definition of a shell) can be omitted. In deMon spherical orbitals are defined over real spherical harmonic Gaussians [27] as (omitting normalization):

$$\phi_{lm}(\mathbf{r}) = r^l \, e^{-\zeta r^2} \, S_l^m \tag{4.5}$$

Here S_l^m are real spherical harmonics which have been constructed from complex spherical harmonics by [28]:

$$S_l^m = \frac{1}{\sqrt{2}} \left(Y_l^m + Y_l^{-m} \right) \tag{4.6}$$

$$S_l^{-m} = -\frac{i}{\sqrt{2}} \left(Y_l^m - Y_l^{-m} \right) \tag{4.7}$$

It should be noted that the S_l^m are not eigenfunction of the $\hat{l}_m$ operator and, therefore, m is no longer a good quantum number for these orbitals.
The ordering of the integer numbers in the configuration line follows the l (shell) and m index of the real spherical harmonics:

Shell	s		p				d			...
l	0		1				2			...
m	0	-1	0	1	-2	-1	0	1	2	...
Orbital	s	p_y	p_z	p_x	d_{xy}	d_{yz}	d_{z^2}	d_{xz}	$d_{x^2-y^2}$	...

As an example the open-shell (OKS or ROKS) triplet ground state configuration of the carbon atom, $1s^2\,2s^2\,2p^2$, can be defined in the keyword block of CONFIGURE as:

```
2 1 1 0
2 0 0 0
```

The first line defines the α-orbital configuration with 2 electrons in the most stable s-orbital ($1s$ and $2s$) and 1 electron each in the most stable p_y ($m = -1$) and p_z ($m = 0$) α-orbital. The second line assigns 2 electrons to the two most stable β s-orbitals. Not specified shells (here d and higher) are not considered. If Cartesian orbitals are used only the number of electrons in the s, p, d, etc. shells have to be specified. Thus, the above configuration changes for Cartesian orbitals to:

```
2 2
2 0
```

Because of the p-orbital degeneracy the chosen Cartesian configuration has to be stabilized during the SCF procedure (see 4.3.4 or 4.4.6). The calculation of the triplet carbon ground state with spherical and Cartesian orbitals using CONFIGURE is described in the examples 5.11 and 5.12, respectively.

In order to access excited atomic states or to use fractional occupation numbers the option OCCUPY has to be selected. In this case an explicit definition of the orbital occupation is expected after the orbital configuration line(s). In the case of spherical atomic orbitals an input line describing the occupation of each real spherical harmonic orbital type is requested. The occupation numbers can be given as real or integer values. To access the excited triplet state of carbon, $1s^2\,2s^1\,2p^2\,3s^1$, the following keyword body of CONFIGURATION, using the option OCCUPY, is needed:

```
3 1 1 0
2 0 0 0
1 0 1
1
1
1 1
```

Here, the `3` in the first configuration line indicates that the three lowest s type α-orbitals have to be occupied according to the occupation scheme given in the third line (`1 0 1`).

Based on this scheme the lowest α s-orbital is occupied with 1 electron, the next is empty and the following one is again occupied with 1 electron. Therefore, a hole in the α s-orbital occupation is produced. The next two occupation lines assign 1 electron to the p_y and to the p_z α-orbital. The two lowest β s-orbitals are occupied with 1 electron each by the last occupation line. Please also note that zero (`0`) entries in the configuration line(s) do not have corresponding occupation lines. Example 5.13 describes the calculation of the excited $1s^2\,2s^1\,2p^2\,3s^1$ carbon triplet state.

As already mentioned the OCCUPY option may also be used to generate fractional occupation, e.g. for the calculation of spherical atoms. In the case of the triplet carbon ground state the following CONFIGURATION keyword body produces a spherical atom:

```
2 1 1 1
2 0 0 0
1 1
0.6666
0.6666
0.6666
1 1
```

The first configuration line describes the occupation of 2 α s-orbitals and of all three p-orbitals. According to the defined occupation (third line) the two s-orbitals are occupied each by 1 electron. The three α p-orbitals are homogeneously occupied by 2/3 of an electron (0.6666 in line 4, 5 and 6). Finally, the β s-orbital occupation is given by the last line. (see Example 5.14).

Because the Kohn-Sham method is a one-determinant approach atomic states have to be approximated by one configuration (see however [29] for a multi-determinant approach). This can be done as in spatially unrestricted Hartree-Fock calculations [30]. For s and p occupations all possible configurations yield the correct spatial symmetry. However, for d occupation this is not the case and care has to be taken to the occupation scheme. The following d configuration have to be used:

$d^1(^2D) : (d_{z^2})^1$
$d^2(^3F) : (d_{z^2})^1\,(d_{x^2-y^2})^1$
$d^3(^4F) : (d_{xy})^1\,(d_{xz})^1\,(d_{yz})^1$
$d^4(^5D) : (d_{x^2-y^2})^1\,(d_{xy})^1\,(d_{xz})^1\,(d_{yz})^1$
$d^5(^6S) : (d_{z^2})^1\,(d_{x^2-y^2})^1\,(d_{xy})^1\,(d_{xz})^1\,(d_{yz})^1$
$d^6(^5D) : (d_{z^2})^2\,(d_{x^2-y^2})^1\,(d_{xy})^1\,(d_{xz})^1\,(d_{yz})^1$
$d^7(^4F) : (d_{z^2})^2\,(d_{x^2-y^2})^2\,(d_{xy})^1\,(d_{xz})^1\,(d_{yz})^1$

$d^8(^3F) : (d_{z^2})^1\,(d_{x^2-y^2})^1\,(d_{xy})^2\,(d_{xz})^2\,(d_{yz})^2$
$d^9(^2D) : (d_{z^2})^1\,(d_{x^2-y^2})^2\,(d_{xy})^2\,(d_{xz})^2\,(d_{yz})^2$

For the d^1, d^4, d^6 and d^9 configurations the orbital occupancies are purely arbitrary, but for the d^2, d^3, d^7 and d^8 configurations the orbital occupancies must be as given above to yield the correct spatial symmetry. Note, however, that for truly multiconfigurational cases, either the ΔSCF method, or time-dependent density functional theory are preferable, and usually lead to more accurate results.

4.4 SCF Control

4.4.1 Keyword SCFTYPE

This keyword selects SCF method and convergence thresholds.
Options:
RKS / **UKS** / **ROKS**

RKS	The restricted Kohn-Sham method will be used. This is the default for closed-shell systems.
UKS	The unrestricted Kohn-Sham method will be used. This is the default for open-shell systems.
ROKS	The spin-restricted open-shell Kohn-Sham method will be used.
MAX=<Integer>	Maximum number of SCF iterations. Default is 100.
TOL=<Real>	Minmax SCF convergence criterion. Default is 10^{-5}.
CDF=<Real>	Fit convergence criterion. Default is $\max(10^{-4}, 10^2 * \mathrm{TOL})$.

Description:
The Kohn-Sham SCF methods are in their implementation similar to the corresponding Hartree-Fock methods [31–35]. Therefore, they have similar advantages and disadvantages. Importantly, in the case of UKS calculations spin-contamination may appear. In deMon the approximated spin contamination [36,37] is calculated and printed before of the converged UKS SCF energy. This value has proven to be a good guide to judge the spin-contamination of UKS calculations. Please keep in mind, however, that this value is calculated for the Kohn-Sham *reference wavefunction*, and not for the true wavefunction of the system. The physical significance of this spin-contamination parameter is therefore not entirely clear.
Due to the variational fitting of the Coulomb potential [38,39] a Minmax SCF procedure is implemented in deMon [40]. In the variational Minmax procedure strict convergence from above is not guaranteed. Therefore, it is possible to obtain energies below the converged energy during the SCF iterations in deMon. Fitting can also lead to small-scale noise (on the order of the SCF convergence criterion) in the energy during the geometry optimization.
The maximum number of SCF iterations is specified with the MAX option. With MAX=0 an "energy only" calculation with the molecular orbital coefficients from the restart file is

performed. No SCF iteration is done! The MAX=0 option automatically triggers GUESS RESTART (see 4.4.4) and, therefore, fails if no adequate restart file `deMon.rst` exists. The ordering and, thus, the occupation of the molecular orbitals in the restart file can be altered with the MOEXCHANGE keyword (see Section 4.3.3).
The SCF convergence criterion can be modified using the TOL option. During geometry optimization, user-defined SCF convergence criterion is applied to the first single point SCF calculation. In the course of optimization, the convergence criterion is automatically tightened according to the residual forces. However, if the user defined convergence criterion is smaller than the actual requested value it is used instead. If self-consistent perturbation calculation is performed the default SCF convergence criterion is increased to 10^{-7}. It may be overwritten by a smaller value with the TOL option, if desired. Note that tight SCF convergence criteria are largerly meaningless, unless tight numerical criteria a used for the the numerical integration of the exchange-correlation potential and ECPs.
In addition to the standard, orbital-based convergence criterion, deMon also monitors the convergence of the fit density. The default convergence criterion is adequate for an overwhelming majority of applications, and **should not be modified**. In a few rare cases (for example, when very accurate values are needed for the multipole moments of the molecular charge distribution), the fit convergence threshold can be modified using the `CDF=` option.

4.4.2 Keyword **ORBIT**ALS

This keyword controls the atomic orbital choice.
Options:
SPHERICAL / **CARTE**SIAN

SPHERICAL	Spherical atomic orbitals ($5d$, $7f$) are used. This is the default.
CARTESIAN	Cartesian atomic orbitals ($6d$, $10f$) are used.

Description:
In deMon the spherical (see 4.3.6 for the definition of real spherical harmonic Gaussians) and Cartesian atomic orbitals have the general form (omitting the normalization):

$$\phi_{lm}(\mathbf{r}) = r^l \, e^{-\zeta r^2} \, S_l^m \tag{4.8}$$

$$\phi_{ijk}(\mathbf{r}) = x^i \, y^j \, z^k \, e^{-\zeta r^2} \tag{4.9}$$

Because spherical orbitals do not include contaminants with lower angular momenta ($l' = l - 2, l - 4, ...$), they are the recommended choice for most applications.

4.4.3 Keyword ERIS

This keyword controls the calculation method for the three-center electron repulsion integrals (ERIs).
Options:
CONVENTIONAL / **DIREC**T / **MULTI**POLE / **MEMOR**Y

CONVENTIONAL	All ERIs are calculated at the beginning of the SCF procedure and stored.
DIRECT	All ERIs are recalculated at each SCF iteration.
MEMORY	Chooses between CONVENTIONAL and DIRECT, based on the available memory. This is the default.
MULTIPOLE	An asymptotic multipole expansion for long-range ERIs is performed. The DIRECT option is activated, too.
TOL=<Real>	Threshold for ERI screening. The default is 10^{-14} for CONVENTIONAL, and 10^{-8} otherwise.

Description:
Calculation of the three-center ERIs is one of the most time critical steps in a deMon calculation. Therefore, the calculation method should be carefully selected. The CONVENTIONAL method calculates ERIs before of the SCF procedure and, if possible, stores them in memory (RAM). This so-called in-core method is extremely fast as long as all integrals fit into the RAM. The RAM size is controlled with the `MAXMEM` keyword (see Section 4.10.1). If the RAM space is not sufficient deMon will write ERIs to the scratch file `ioeri.scr`. Thus, the ERIs have to be read from the disk in each SCF step. For larger systems this disk I/O becomes the bottleneck of the calculation.
The DIRECT method [41] avoids the disk I/O by recalculating the ERIs in each SCF step. This is, therefore, the method of choice for larger systems where the ERIs do not fit in the available RAM space.
The default MEMORY method will choose between a CONVENTIONAL and DIRECT calculations, based on the amount of the available memory (see 4.10.1). This method will typicall lead to the lowest wall-clock time for a calculations. On computer systems with

an unusually high I/O capacity, or when CPU time, rather than wall-clock time is of interest, CONVENTIONAL approach may be preferable.
If the calculated system has a large extension ($\geq$ 10 Å) asymptotic expansions for long-range ERIs [42] can be used. This method is activated with the MULTIPOLE option. Because asymptotic expansions are only implemented in the DIRECT method the MULTIPOLE option also triggers the DIRECT option. This is the method of choice for large extended systems.
Table 7 shows timings for the ERI calculation of small alkenes with the three methods. The DZVP basis and A2 auxiliary function set was used. The SCF convergence (10^{-5}) was reached for all systems within 10 cycles. Thus, the ERIs hat to be calculated ten times in the DIRECT and MULTIPOLE method. Table 7 shows the total time for these two methods and the time for the (one) ERI calculation with the CONVENTIONAL method. In brackets the difference between the real and CPU time is given, too. All calculations were performed on a single SGI R14000 node with 2 Gbyte of RAM and a Ultra SCSI disk.

Table 7: Timing [sec] for the ERI calculation of alkenes with the CONVENTIONAL, DIRECT and MULTIPOLE method. The number of basis and auxiliary functions is given by N_{Basis} and N_{Auxis}. The difference of real and CPU time is given in brackets.

Alkene	N_{Basis}	N_{Auxis}	CONVENTIONAL	DIRECT	MULTIPOLE
$C_{24}H_{50}$	610	1016	132 (14)	345 (5)	266 (5)
$C_{36}H_{74}$	910	1520	306 (220)	862 (434)	567 (300)
$C_{48}H_{98}$	1210	2024	572 (2374)[a]	1522 (493)	898 (264)
$C_{60}H_{122}$	1510	2528	914 (2516)[a]	2375 (465)	1294 (255)
$C_{72}H_{146}$	1810	3032	1297 (2558)[a]	3428 (466)	1762 (260)

[a] Disk I/O is performed.

As Table 7 shows the CONVENTIONAL method is most economic for the two smallest alkenes. In these cases in-core calculations can be performed and the disk I/O is reduced to a minimum. The ERI calculation with the DIRECT method takes almost three times more. Remarkable is also the relative large overhead of the DIRECT method (434 sec) for $C_{36}H_{74}$. The reason is the incremental build of the Kohn-Sham matrix which involves I/O operations (reading and writing of the Kohn-Sham matrix) not present in the in-core calculation. However, the Kohn-Sham matrix is only a N^2_{Basis} object and, therefore, the

I/O overhead of the DIRECT method becomes less important for large systems.
In the case of the $C_{48}H_{98}$ and larger alkenes the ERIs do not fit any longer in the RAM. Now I/O ERI operations are necessary with the CONVENTIONAL method. This immediately increases the overhead of the calculations. Altogether, the real time for the CONVENTIONAL calculation in $C_{48}H_{98}$ (2946 sec) is larger than for the DIRECT calculation (2015 sec). A further reduction is obtained with the MULTIPOLE method (1162 sec). As can be seen from Table 7 the MULTIPOLE method becomes more favorable with increasing size of the alkenes. This is due to the reduced scaling ($N_{Basis}^{1.7}$) of this method even for such small systems as the alkenes in Table 7.
The TOL option controls screening of the ERIs. The screening threshold τ is calculated as:

$$\tau = \frac{\text{TOL}}{\text{Number of Electrons}} \tag{4.10}$$

ERIs with an orbital overlap smaller than τ are not calculated (screened). The threshold τ also enters into the asymptotic expansion radii for the MULTIPOLE method. The density screening threshold for the numerical integration of the exchange-correlation potential is given by τ, too. The default settings for TOL are 10^{-14} and 10^{-8} for the CONVENTIONAL and DIRECT/MULTIPOLE method, respectively. Please note that the screening of the ERIs can deteriorate the SCF convergence, particularly for systems with extensive electron delocalization.

4.4.4 Keyword GUESS

This keyword specifies the SCF start density.
Options:
TB / **CORE** / **FERMI** / **RESTA**RT

TB	A tight-binding SCF start density is calculated. This is the default.
FTB	Use “Free” tight-binding guess. This is a synonym for TB.
HTB	Use “Harmonic” tight-binding guess.
CORE	The SCF start density is obtained from the diagonalization of the core Hamiltonian.
FERMI	The start density is obtained by quenching a fractional occupied SCF solution to integer occupation numbers.

RESTART	The start density is read from the restart file `deMon.rst`.
ONLY	The program stops after the generation of the start density.
HARM=	Radius of the harmonic restraining potential, in units of covalent atomic radius. The default is 2.0
FREE=	Radius of the restraining potential for a "free" atom. The default is 100.0

Description:
The choice of the start density can be crucial for the SCF convergence. In most cases the tight-binding start density is the recommended choice. For metal clusters the core start density may be sometimes advantageous. If the molecular orbitals of the start density (for the printing of the MOs see Section 4.10.3) show only a very small HOMO-LUMO gap the FERMI option may be used to obtain a better start density. It should be noted that this option implies a SCF calculation and, therefore, is much more time consuming than the other start density options. If a FERMI start density is used the OMA option of the keyword MIXING (see 4.4.5) should not be applied! The start density can also be read from the restart file `deMon.rst` of a previous run. With the keyword MOEXCHANGE (see 4.3.3) the molecular orbital ordering of the start density can be altered. The option ONLY stops the program after the start density is generated and written to the restart file. This option is recommended if a FERMI start density was requested or the start density should be altered after inspection (see example 5.14). Note that `GUESS RESTART` is currently not supported for geometry optimizations. To print guess molecular orbitals, use a combination of the keywords `SCFTYP MAX=1` and `PRINT MOS`.

4.4.5 Keyword MIXING

This keyword controls the charge density mixing.
Options:

+<Real>	Fixed charge density mixing parameter between 0 and 1
-<Real>	Dynamic charge density mixing parameter. The default is -0.3.
OMA	The optimal mixing algorithm is activated.

Description:
Hartree damping [44] of the charge density fitting coefficients is performed, using the specified the charge density mixing parameter. Smaller mixing parameters correspond to

stronger damping. For some systems mixing parameter smaller than default (e.g. 0.1) may be necessary to achieve convergence. If a dynamic mixing parameter is chosen, the mixing ratio is reduced during the SCF iterations if the Minmax SCF error increases. The OMA option activates the optimal mixing algorithm of Cancés [43]. This option is recommended if the dynamical mixing fails or is converging slowly. It should be noted that OMA possesses a considerable overhead and exhibits bad convergence for small HOMO-LUMO gaps.

4.4.6 Keyword SHIFT

This keyword activates the level-shift procedure.
Options:

+<Real>	Fixed level-shift value [in a.u.].
-<Real>	Dynamic level-shift value [in a.u.].

Description:
The level-shift procedure allows artificially expanding the HOMO-LUMO gap during SCF iterations [45]. This stabilizes the initial SCF configuration. The shift value is subtracted from the orbital energies once SCF procedure has converged. Because of the small HOMO-LUMO gap in DFT calculations this procedure is very valuable to improve SCF convergence. However, with a fixed level-shift calculation may converge to an excited state. Therefore, the converged orbital occupation should be examined using the MOS option of the PRINT keyword (see 4.10.3). This problem can be reduced if the dynamical level-shift procedure [14] is used. Here the shift value is adapted to the Minmax SCF error. While the error is large, usually at the beginning of the SCF procedure, the full shift value is used. When the error decreases the shift value is decreased, too (see Example 5.15). Usually a small shift value remains at the end of the SCF procedure. The dynamical level-shift procedure has proven very valuable for the SCF convergence of transition metal clusters and systems [46].
If the SHIFT keyword is used, the level-shift procedure is applied in all calculation steps (optimization, frequencies, properties, etc.). If this is not desirable, a single point SCF can to be performed with the SHIFT keyword. The converged density can then to be used as the restart density (see 4.4.4) in the following calculations without the SHIFT keyword (See example 5.16).

A cautionary note: Neither having the aufbau orbital occupations at the end of the SCF, nor using dynamis level shifting procedure provide a *guarantee* that your calculation converged to the ground electronic state. Conversely, in approximate density-functional theory it is also possible (although extremely rate) to have a non-aufbau ground state. Caveat user!

4.4.7 Keyword DIIS

This keyword activates the DIIS procedure.
Options:
ON / **OFF**

ON	The DIIS procedure is switched on. This is the default.
OFF	The DIIS procedure is switched off.
TOL=<Real>	The DIIS procedure is switched on after the SCF energy error is smaller than <Real>.
CMAX=<Real>	Largest allowed expansion coefficient. Default is 100.0
DMIN=<Real>	Smallest allowed determinant. Default is 10^{-14}

Description:
By default the DIIS (direct inversion in the iterative subspace) procedure [47] is switched on if the energy error is less than 0.1 a.u. in a SCF step. For most systems this result in a considerable speed-up of the SCF convergence. However, for some systems this threshold is too large and a SCF convergence failure will occur. In these cases the start of the DIIS procedure should be manipulated with the TOL option. If the HOMO-LUMO gap of a system is very small DIIS can be counter productive for the SCF convergence. In these cases the DIIS procedure can be switched off with the OFF option. It should be noted that in deMon different DIIS algorithms are used for the AUXIS and BASIS option of the VXCTYPE keyword (see 4.4.8). Both are based on the energy gradient with respect to the charge density fitting coefficients. In the case of the AUXIS option only charge density fitting coefficients are used in the DIIS step. Therefore, no extra I/O is necessary. In the case of the BASIS option also density matrices are used in the DIIS step, leading to an increase in the I/O overhead.
DIIS procedure will be restarted if any of the extrapolation coefficients exceeds CMAX in absolute magnitude, or if the determinant of the DIIS linear problem drops below

DMIN. The defaults for these values are adequate in an overwhelming majority of cases, and should not be tinkered with.

4.4.8 Keyword VXCTYPE

With this keyword the exchange-correlation potential is selected. It also controls the density used for the exchange-correlation energy and potential calculation.
Options:
VWN / **PZ81** / **PW92** / **PW86** / **BLYP** / **PW91** / **PW91S**SF / **PBE** / **PBESS**F / **XALPH**A / **NONE**

VWN	Dirac exchange with local VWN correlation.
PZ81	Dirac exchange with local PZ81 correlation.
PW92	Dirac exchange with local PW92 correlation.
PW86	PW86 GGA exchange with P86 GGA correlation.
BLYP	B88 GGA exchange with LYP GGA correlation.
PW91	PW91 GGA exchange and correlation.
PW91SSF	PW91 with full spin scaling function.
PBE	PBE GGA exchange and correlation.
PBESSF	PBE with full spin scaling function.
XALPHA	X_α calculation. The default α value is 0.75. A user defined α value can be selected with the X = <Real> option.
NONE	No exchange-correlation functional is used.

(ie Hartree calculation is done).

AUXIS / **BASIS**

AUXIS	The auxiliary function density is used for the calculation of the exchange-correlation energy and potential. This is the default.
BASIS	The orbital density is used for the calculation of the exchange-correlation energy and potential.

Description:
The above options represent the most common combinations of exchange and correlation functionals. Besides these combinations the exchange and correlation functionals listed in Table 8 can be combined by the user. The syntax for the user defined potentials is

<Exchange> - <Correlation> (e.g. B88 - P86). With NONE the exchange or correlation part can be omitted. In contrast to many examples in the literature, the PW86 exchange functional in deMon is implemented with a cutoff. The local contribution of the P86 correlation functional is calculated using the VWN functional. All other functionals are implemented according to the cited references. Where possible, second derivatives with respecy to density were eliminated by integration by parts [48].
The AUXIS and BASIS options specify the density that is used for the calculation of the exchange-correlation energy and potential. By default the auxiliary function density is used for the calculation of the exchange-correlation energy and potential. Using the option BASIS results in a significant slow-down of the SCF calculation. However, for sensitive property calculation the BASIS option is recommended. For more details and recommendations see Section 1.4, *"How to Use deMon"*.
For the PW91 and PBE functional two different spin scaling functions are implemented in deMon. By default a numerical more stable cutoff function is used. The suffix "SSF" (options PW91SSF and PBESSF) selects the original form of the spin scaling function. This may change the orbital energies considerably. For the total energies the effect is usually negligible. For other functionals, the default choice of the spin scaling function can be overriden by specifying `SSF=FULL` or `SSF=G98` options of the VXTYPE keyword.
Note: In the literature, combinations of the `PBE` correlation functional, and the `PBE98` and `PBE99` exchange functionals are often refered to as "revPBE" and "RPBE" functionals, respectively.

4.4.9 Keyword GRID

This keyword specifies the grid for the numerical integration of the exchange-correlation energy and potential.
Options:
ADAPTIVE / **FIXED**

ADAPTIVE	An adaptive grid is used for the integration. This is the default.
FIXED	A fixed grid is used for the integration.

MEDIUM / **COARS**E / **FINE**

MEDIUM	A medium grid accuracy is requested. This is the default.
COARSE	A coarse grid accuracy is requested.

Table 8: Exchange and correlation functionals available in deMon. The minimal abbreviations are given in bold.

Exchange:	
DIRAC	Local Dirac exchange [49]
PW86	Perdew and Wang GGA exchange from 1986 [50]
B88	Becke GGA exchange from 1988 [51]
PW91	Perdew and Wang GGA exchange from 1991 [52]
PBE96	Original Perdew, Burke and Ernzerhof GGA exchange from 1996 [53]
PBE98	PBE GGA exchange modified by Y. Zhang and W. Yang [54]
PBE99	PBE GGA exchange modified by B. Hammer et al. [55]
Correlation:	
VWN	Local Vosko, Wilk and Nusair correlation [56]
PZ81	Local Perdew and Zunger correlation [57]
PW92	Local Perdew and Wang correlation from 1992 [58]
PZ86	Perdew GGA correlation from 1986 [59]
P86	Perdew GGA correlation from 1986 with VWN local correlation [60]
LYP	Lee, Yang and Parr GGA correlation from 1988 [61–63]
PW91	Perdew and Wang GGA correlation from 1991 [52]
PW91SSSF	PW91 with full spin scaling function
PBE	Perdew, Burke and Ernzerhof GGA correlation form 1996 [53]
PBESSF	PBE with full spin scaling function

FINE A fine grid accuracy is requested.

TOL=<Real> Adaptive grid tolerance.

SCF / GUESS

SCF The converged SCF density is used for the adaptive grid generation. This is the default.

GUESS The start density is used for the adaptive grid generation.

Description:

By default, deMon uses an adaptive grid [64] with a tolerance of 10^{-5} (MEDIUM) for the numerical integration of the exchange-correlation energy and potential. This tolerance corresponds to the accuracy of the numerical integration of the diagonal elements of the exchange-correlation potential matrix. For the converged SCF energy an accuracy better

than 100 μHartree is usually obtained with this setting [65]. The COARSE and FINE option for the adaptive grid refer to grid tolerances of 10^{-4} and 10^{-6}, respectively. Thus, the stability of the numerical integration can be easily checked by choosing different grid tolerances. The COARSE adaptive grid should not be used for final energy or property calculations. User-defined grid tolerances can be specified using TOL option. This option is only applicable for an adaptive grid. The same holds for the SCF and GUESS options, which specify the trial density used for the grid generation. By default, the converged SCF density is used. This implies a second SCF cycle for the initial energy calculation. If the option GUESS is specified the adaptive grid is build using the start density for the calculation of the grid generating exchange-correlation potential.
Molecular dynamics simulations, where a good guess density is tipically available on each step, typically run faster with `GUESS` grids, due to smaller computational overhead due to grid generation.
Fixed grids available in deMon can be selected by the option FIXED in combination with the options COARSE, MEDIUM and FINE, with MEDIUM being the default. Options COARSE and MEDIUM select (50,194)p and (75,302)p pruned grids, respectively. In this notation the first value refer to the radial shells. The second value is the size of the largest Lebdev grid [67] on each of these shells. The p stands for pruned [66] indicating that smaller Lebedev grids have been used for some radial shells. The FIXED FINE option requests the unpruned (200,1202) reference grid. Due to its size its application is impractical for all but smallest systems.

4.5 Molecular mechanics and QM/MM Control

4.5.1 Keyword QMMM

This keyword activates molecular mechanics or hybrid QM/MM calculation.

Options:

QM / **MM** / **QM/MM**

QM	Treat the whole system using quantum mechanics; parse but ignore all MM- and QM/MM-related input keywords. This is the default.
MM	Treat the whole system using molecular mechanics.
QM/MM	Partition system into QM and MM parts, as specified by the QM/MM keywords in the geometry section. This keyword can also be given as simply **ON**.
COUPLING=	Specifies coupling Hamiltonian, used to describe interactions between the QM and MM parts. The default is `MECHANICAL`.
RUNTYPE=	Specifies time synchronization between the partitions. The default is `SYNCHRONOUS`.

Description:

With the default `QMMM QM` setting, all keywords and flags related to molecular mechanics and hybrid calculations are recognized, but ignored in the input file. The whole system is therefore treated using DFT Hamiltonian. The opposite setting, `QMMM MM` will use molecular mechanics forcefield (specified by the `FORCEFIELD` keyword, see 4.5.2) to treat the entire system. If this option is specified, deMon will behaive as if `QMMM=MM` has been given for each atom in the `GEOMETRY` section (see 4.1.1). This option is particularly useful for the initial geometry optimization, or for the initial part of a molecular dynamics equilibration run. Finally, the `QMMM QM/MM` will honour all QM/MM-related input options.

The two options supported for the QM/MM coupling Hamiltonian (`COUPLING=`) are:

`MECHANICAL`	Use simple mechanical embedding. The total energy of the system in this case is given by: $E_{tot} = E_{MM}(QM + MM) - E_{MM}(QM) + E_{QM}(QM)$. Apart from bond termination, the molecular mechanics subsystem does not directly influence the electronic structure of the QM part.
`ELECTROSTATIC`	Also include cross-system electrostatic polarization terms. This option is not yet implemented.

Regardless of the coupling scheme used, bond termination at the partition boundary is

handled using capping effective potential approach, using either pseudo-halogen[10] or pseudo-hydrogen[11] pseudopotentials.

The dynamical relationship between the QM and MM partitions can be controlled with the `RUNTYPE=` keyword. The recognized options are:

`SYNCHRONOUS` Time steps are identical across the partitions. This is the default.

`RELAX_MM` The MM partition is completely relaxed for every time step in the QM partition. This option is not yet implemented.

`FREE_ENERGY` Time runs faster in the MM partition; the QM partition moves on the free-energy surface of the MM partition. This option is not yet implemented.

If your application requires a currently unimplemented coupling scheme or RUNTYPE, you are encouraged to contact: `Serguei.Patchkovskii@nrc.ca`.

Note: Currently, the `RESTART` option (see **??**) is not supported for calculations where a change in the system Hamiltonian occurs. Molecular dynamics simulations can be restarted (see **??**) from a simulation using different system partitioning.

4.5.2 Keyword FORCEFIELD

This keyword is used to specify molecular mechanics forcefield, and to set forcefield-related computational options. Options:

TYPE= Chooses molecular mechanics forcefield. This can be either `UFF` or `EXTERNAL`. Only **UFF**, which is the default, is currently implemented.

BONDS= Chooses the bond matrix definition. This can be either `EXPLICIT` or `AUTO`. `AUTO` is the default.

PROG= Name of an external molecular mechanics program. There is no default.

Description:

The default molecular mechanics forcefield, used in molecular mechanics and QM/MM calculations is the Universal Force Field (UFF) by Rappe et al.[68–73] (see also section 6). While this force field is not particularly accurate for any specific system, it covers the entire periodic table, and often produces *reasonable* structures. This forcefield implementation is integrated in deMon, and carries minimal overhead in applications.

The option `BONDS=` selects handling of the connection matrix, which is needed in most MM force fields. The two options are:

`EXPLICIT` All bonds and bond orders must be specified explicitly, in the geometry input (see keywords `BOND=` and `NOBOND=` in section 4.1.1.

`AUTO` deMon will try to guess the connection matrix, suitable for the specified starting geometry. It is still possible to override the guess (partially or completely) by supplying `BOND=` and `NOBOND=` flags in the GEOMETRY section. This is the default.

Finally, `PROG=` chooses the name of the external executable (or a shell script), which will be called to evaluate MM energies and forces, if an external force field is activated. This must be a absolute path name - paths relative to the current directory will not work. There is no default for this option.

4.5.3 Keyword MMOPTIONS

This keyword sets various technical parameters for the MM simulation part. Not all of those parameters have a meaning for all run types.

Options:

TSTEP= MM MD time step, in atomic time units.

STEPS= Number of MM MD time steps, or maximum number of MM geometry optimization steps.

TOLE= Energy convergence criterion for MM optimizations, in Hartree.

TOLG= Gradient convergence criterion for MM optimizations, in Hartree/Bohr.

TOLX= Displacement convergence criterion for MM optimizations, in Bohr.

TOL= Set TOLE, TOLG, and TOLX to values suitable for energy convergence specified here, in Hartree.

TAU= Berendsen tau parameter, in atomic time units.

T= MM simulation temperature, in Kelvin.

4.5.4 Keyword VDWAALS

This keyword sets defaults for van-der-Waals non-bonded interaction summation cutoffs (used in DFTB, MM, and QM/MM).

Options:

ECUT= Energy cutoff in Hartree. The default is 10^{-6}.

Description:

All van-der-Waals type non-bonded pair interactions smaller than `ECUT` will be neglected.

This parameter affects both periodic and non-periodic calculation. Although this cut-off looks extremely tight, it is necessary to guarantee total energy conservation to within 0.1 kcal/mol in a typical periodic MD simulation.

4.5.5 Keyword MADELUNG

This keyword sets defaults for the summation of the Madelung contribution to the electrostatic potential (used in DFTB, MM and QM/MM). Options:

D= d parameter of the screening function, in Bohr. The default is 6.0

R0= r_0 parameter of the screening function, in Bohr. The default is 70.0

ECUT= Energy cutoff, in Hartee. The default is 10^{-6}

Description:

In periodic MM, QM/MM, and DFTB calculation on systems with non- negligible atomic charges, deMon must evaluated the electrostatic potential of the infinite periodic charge distribution (Madelung potential). The approach currently implemented in deMon employs real-space summation, with a screened interaction potential[74]. In this technique, the usual Coulomb interaction potential is replaced with:

$$v^{\mathrm{scr}}(r) = \frac{1}{r} * \frac{1}{1 + \exp\left((r - r_0)/d\right)} \tag{4.11}$$

The screening parameters r_0 and d must be selected such that:

- The screening function is not distinguishable from 1 within the unit cell surrounding the origin.
- The fall-off region of the screening function is wider than the typical extent of non-neutral charge distrubutions in the system.

The defaults are appropriate for simulation of liquids of small polar molecules. Simulations of systems with long-range charge correlations may require large values of d and r_0.

4.5.6 Keyword PEDIOIC

This keyword activates periodic boundary conditions in MM and QM/MM calculations, and specifies the initial extent of the unit cell.

Options:

CUBIC / **RECTA**NGULAR / **GENER**AL

CUBIC	Cubic simulation box is used.
RECTANGULAR	Rectangular simulation box is used.
GENERAL	General simulation box is used. This is the default.

Description:

The `PERIODIC` keyword requires additional input, which must follow it in the input, with no intervening comments or blank lines. For the `CUBIC` box, just one parameter is required - the extent of the box edge, in Bohr. For the `RECTANGULAR` box, the extents along the X, Y, and Z spatial directions must be specified. In either case, the lattice vectors are taken to be parallel to the coordinate axes: $a||X$, $b||Y$, and $c||Z$. For a `GENERAL` unit cell, the Cartesian coordinates of the a, b, and c lattice vectors must be specified, with one vector per line.

Note that in a molecular dynamics run, the unit cell parameters may be affected by the pressure bath (see keyword `MDPRESSURE` in section 4.7.8).

4.6 Optimization Control

4.6.1 Keyword OPTIMIZATION

This keyword controls the geometry optimization and transition state search.
Options:
REDUNDANT / **INTER**NAL / **CARTE**SIAN

REDUNDANT	The optimization is performed in delocalized redundant internal coordinates. This is the default.
INTERNAL	The optimization is performed in the specified internal coordinates (Z-Matrix input).
CARTESIAN	The optimization is performed in automatically generated redundant internal coordinates.
MAX=<Integer>	Maximum number of optimization steps. Default is 50.
TOL=<Real>	Optimization convergence criterion. Default is 3×10^{-4}.
STEP=<Real>	Maximum step size in optimization. Default is 0.3 Bohr
TS	Activate transition state search.
MOD=<Integer>	Hessian eigenmode to be followed in the transition state search. Default is 1, the mode with the lowest (most negative) frequency.

Description:
By default, deMon uses delocalized redundant internal coordinates for the geometry optimization and transition state search [75]. For full geometry optimizations this is also the recommended method. However, equivalent coordinates are currently not supported for redundant coordinates. Therefore, the INTERNAL option in combination with a Z-Matrix input (see 4.1.1) has to be used in this case. Because of the possible linear dependencies of the internal coordinates in the Z-Matrix, this optimization may become problematic for larger systems ($\geq$ 30 atoms). With the CARTESIAN option deMon constructs a Z-Matrix from the Cartesian input avoiding linear dependencies. This option should be used if an optimization gets stuck and has to be restarted.
Note that the geometry optimizer, invoked by the OPTIMIZATION keyword, is computationally expensive, and is not suitable for large QM/MM calculations. For situations where energy evaluation is inexpensive, while the number of degrees of freedom is large,

simulated annealing protocol of the molecular dynamics part of deMon will usually perform better for geometry optimization.
The maximum number of optimization steps and the optimization convergence criterion can be specified using the MAX and TOL options. The convergence criteria are based on the remaining and root mean square (rms) of the residual force. The TOL option specifies the criterion for the rms force. Other convergence criteria are derived from this value. The maximum step length (in atomic units) used in the optimization can be specified with the STEP option.
Option TS requests a transition state search based on the eigenvector following for the Hessian matrix. The starting structure and the initial Hessian are crucial for the success of this approach. It is recommended to explicitly calculate the initial Hessian (see Section 4.6.2) for the transition state search. In deMon (P)-RFO steps [76,77] are taken in the transition state search. This guarantees that the right Hessian structure (one negative eigenvalue) is obtained or preserved. In order to avoid retention to a positive definite Hessian the Powell update [78] is used by default for the transition state search.
If other than the lowest Hessian eigenmode should be followed, option MOD can be used to select the desired eigenvector (use the PRINT keyword 4.10.3 with the DE2 option to print the Hessian eigenvectors). On subsequent steps this mode is selected according to the largest overlap with the eigenvector followed in the previous cycle.

4.6.2 Keyword **HESSI**AN

With this keyword the start Hessian for the geometry optimization or transition state search is selected.
Options:
BAKER / **FISHE**R / **LINDH** / **CALCU**LATE / **RESTA**RT / **UNITY** / **READ**

BAKER	Start Hessian constructed from Baker force weights. This is the default for the internal optimization.
FISHER	Start Hessian constructed from Fisher force weights. This is the default for the redundant optimization.
LINDH	Start Hessian constructed from Lindh force weights.
CALCULATE	The start Hessian is calculated.
RESTART	The start Hessian is read from the restart file `deMon.rst`.
UNITY	The unit matrix is used as start Hessian.

READ The start Hessian is read from the input file.

Description:
If options BAKER, FISHER and LINDH are specified, initial Hessian is estimated in primitive redundant coordinates and then transformed to the coordinate system used for the optimization. With the option BAKER the diagonal elements for the primitive bonds, bends and dihedral angles are set to 0.5, 0.2 and 0.1 respectively [75]. With the option FISHER the diagonal elements are calculated according to the empirical rules of Fisher et al. [79]. FISHER is the default for the start Hessian of the redundant internal optimization. With the option LINDH the diagonal elements are calculated according to the empirical rules of Lindh et al. [80].
A unit matrix in the space of the primitive redundant coordinates (for redundant optimizations) or in the space of the internal Z-Matrix coordinates (for internal optimizations) is used with the option UNITY. The calculation of the start Hessian, using numerical second derivatives, is triggered by the option CALCULATE. This is the most accurate, but also time consuming, method. It is recommended for the transition state search. With the options RESTART and READ the start Hessian is read from the restart or input file. The (updated) Hessian matrix is written to the restart file at the end of the optimization and also after it has been calculated using the HESSIAN or FREQUENCY keyword (see 4.8.5). With the RESTART option it can be recovered. If the READ option is specified, the elements of the square Hessian matrix have to be given row-by-row in free format as real values in the keyword body of the HESSIAN keyword (see Example 5.17).

4.6.3 Keyword **STEPT**YPE

With this keyword the geometry optimization step is selected.
Options:
RFO / **WALK** / **DESCE**NT

RFO	Rational function optimization.
WALK	Potential energy surface walking.
DESCENT	Steepest descent.

Description:

The RFO [76] and WALK [81] steps are both restricted step methods [82]. For the transition state search the (P)-RFO step is used regardless of the STEPTYPE specified in the input (see 4.6.1).

4.6.4 Keyword UPDATE

This keyword controls the Hessian update method during geometry optimization or transition state search.
Options:
BFGS / **POWELL** / **PBFGS** / **BERNY** / **DFP** / **MSP**

BFGS	Broyden, Fletcher, Goldfarb, Shanno (BFGS) update [83–86]. This is the default for the geometry optimization.
POWELL	Powell update [78]. This is the default for the transition state search.
PBFGS	BFGS update with numerical threshold in order to preserve a positive definite Hessian matrix.
BERNY	Berny update from H.B. Schlegel [87].
DFP	Davidon, Fletcher, Powell update [88,89].
MSP	Murtagh-Sargent-Powell update [90].

Description:
For the optimization in redundant coordinates the BFGS and PBFGS updates are recommended. For the optimization in internal coordinates the BERNY update may be preferred. For the transition state search only the POWELL or MSP update are suitable.

4.7 MD Control

4.7.1 Keyword MDYNAMICS

This keyword activates the Born-Oppenheimer molecular dynamic (BOMD) simulation.
Options:

ZERO / **RANDO**M / **RESTA**RT / **SDC** / **SDC=**<value> / **RESET**

RESTART	Restart previous molecular dynamics simulation from `deMon.qmd` file
ZERO	Start a new MD run, witht the initial velocities of the nuclei set to zero. This is the default.
RANDOM	Start a new MD run, using random initial velocities
READ	Start a new MD run. Velocities of the nuclei are given in the input file, immediately following the `MDYNAMICS` keyword.
SDC	A steepest descent geometry optimisation is performed.
SDC=<value>	A steepest descent optimisation is performed. After all forces are lower than <value> (in atomic units) the optimisation terminates.
RESET	This option resets all averages and the MD step of the trajectory. It makes only sense with the RESTART option, otherwise it has no effect.

Description:
During an MD restart, the following processing occurs: Atom types and masses are taken from the input file. Coordinates, velocities, and (for periodic calculations) unit cell parameters are taken from the MD restart file, which has a `.qmd` extension. Running averages of the temperature, total and potential energy, pressure, etc. are taken from the MD restart file as well. All other computational parameters (SCF, MD, QM/MM, etc) are taken from the input file. If the molecular dynamics time step or QM/MM partitioning scheme are different in the restart file and the input file, all running averages will be reset, and simulation will start at time $t = 0$. **WARNING:** You should always use the same atom ordering in the input file used to produce the MD restart file, and in the file used to contue the simulation. In particular, you should **never use the `.new` file generated by deMon** for performing an MD restart – there is no guarantee that atom ordering is identical in the original input file and the `.new` file.
When `MDYNAMICS RANDOM` is specified, the initial velocity components of an atom i are calculated as:

$$
\begin{aligned}
v_x(i) &= k\frac{4}{m_i}\sin\left(2\pi(rand-0.2)\right) \\
v_y(i) &= k\frac{4}{m_i}\sin\left(2\pi(rand-0.2)\right) \\
v_z(i) &= k\frac{4}{m_i}\sin\left(2\pi(rand-0.2)\right)
\end{aligned}
$$

where *rand* is a random number, uniformly distributed on the interval $[-1 : +1]$, and m_i is mass of atom i. Scaling coefficient k is calculated such that the initial kinetic energy matches the desired temperature T. **Important:** this initial velocity distribution is non-physical, and must be equilibrated before any production sampling runs.
`MDYNAMICS READ` expects velocity specification in the format:

```
  I    VX    VY    VZ
```

where `I` is the sequential atom number (in the input order), and `VX`, `VY`, and `VZ` are initial velocity components. Velocities must be specified in Bohr/au[time] or in Å/femtosecond, depending on the units used to specify the geometry. Atoms not mentioned in `MDYNAMICS READ` input section are given zero initial velocities.

4.7.2 Keyword MDSTEPS

This keyword controls the steps of the molecular dynamic (MD) simulation.
Options:

MAX=<Integer>	Maximum number of MD steps. The default is 1, for `SDC` 100.
TMAX=<Real>	Maximum MD run time in ps.
OUT=<Integer>	Step interval to update the `deMon.mol` file. The default is 10.

Description:
During the molecular dynamics run the geometry, potential energy, kinetic energy, total energy and temperature will be written to the file `deMon.mol`. The option OUT specifies the step interval after which a new entry is written to `deMon.mol`. This file also contains the average values of the energies and temperature calculated over the total simulation.

4.7.3 Keyword MDTEMP

This keyword specifies the desired temperature of the molecular dynamic (MD) simulation.

Options:

<Real> Desired MD temperature in Kelvin. The default is 300.

Description:
This keyword may affect a molecular dynamics run in two ways. For `MDYNAMICS RANDOM`, the desired temperature is used to rescale initial velocities. Otherwise, in a normal MD run, `MDTEMP` has no effect unless one of the thermostat options is activated - see `MDBATH` (section 4.7.5).

4.7.4 Keyword **TIMES**TEP

This keyword specifies the time step of the molecular dynamic (MD) simulation.
Options:

<Real> MD time step in femtoseconds [fs]. The default is 1

Description:
The default MD time step is determined by the most light weighted atom and the simulation temperature and rounded down to 1/4 fs units. The determination is done using the following formula: $dt = 0.25 * \sqrt{m_{min}/m_H} * \sqrt{300K/MAX(T_0, 5K)}$. This assumes a time step of 0.25 fs for a system containing protons at 300K. In systems where hard wall-like parts of the potential energy surfaces are probed (e.g. gases at high pressures), time steps smaller than the default may be required to maintain total energy constant. An excessively large time step will manifest itself in poor energy conservation during an MD simulation.
Negative time steps allow reverse-time dynamics. However, note that this option is extremely sensitive to numerical errors, which include time steps, SCF convergence and alike.

4.7.5 Keyword **MDBAT**H

This keyword specifies the temperature bath used in the molecular dynamic (MD) simulation.
Options:
NONE / **SCALI**NG / **BEREN**DSEN / **LOCAL**

NONE	No temperature bath. This is the default.
SCALING	The velocities are scaled in order to control the temperature.
BERENDSEN	Berendsen thermostat is applied to the system as a whole.
LOCAL	Berendsen thermostat is applied individually to each atom.
VAL=<Integer>	`SCALING`: number of steps between velocity resets.
VAL=<Real>	`BERENDSEN` or `LOCAL`: Thermostat time constant τ in picoseconds [ps]. Default is $\tau = 0.5$ ps.

Description:

Free-running MD equations generate trajectories with constant energy, to within the numerical error of the time evolution algorithm (velocity Verlet in deMon). In other words, molecular dynamics simulation is performed for the microcanonical (N,V,E) ensemble. However, it is often of interest to perform simulations at constant temperature, which requires modifications to the standard equations of motion. There are a number of different approaches for performing constant temperature MD.

In deMon, the temperature can be controlled by scaling the velocities, i.e. at each time step the velocities are scaled according to $v' = kv$. The three temperature control methods, implemented in deMon, differ in how this constant k is calculated.

With the `SCALING` thermostat, the velocities of all atoms are scaled (by the same factor), so that the instantaneous temperature T_{kin} matches desired temperature T_0 set by the MDTEMP keyword (see Section 4.7.3). In this case k is simply:

$$k = \sqrt{\frac{T_0}{T_{kin}}} \tag{4.12}$$

If the Berendsen thermostat [91] is used the scaling factor k is calculated using the Berendsen parameter τ and the time step, desired temperature and instantaneous temperature from the dynamics.

Because rescaling introduces a large perturbation in system's dynamics, `MDBATH SCALING` should only be used for very rough, initial equilibration of the system.

With `BERENDSEN` thermostat [91], the scaling coefficient is calculated as:

$$k = \sqrt{1 + \frac{\delta t}{\tau}\left(\frac{T_0}{T_{kin}} - 1\right)} \tag{4.13}$$

where δt is the MD time step. That is, the average temperature in this case will gradually approach the desired simulation temperature, with a characteristic time constant τ.

Finally, `MDBATH LOCAL` invokes local version of the Berendsen thermostat. In this case, scaling coefficient k is still given by eq. 13. However, it is now calculated *per-atom*,

from the individual atomic kinetic energies. As a result, local Berendsen thermostat will approach equilibrium faster that the global version, at the cost of introducing additional dynamical aftefacts in the system.
Typically, a molecular dynamics simulation would use `SCALING` thermostat for the initial equilibration, then complete the equilibration with local Berendsen thermostat. The production runs, used to collect statistical information on the system then will be done using either free-running dynamics, or global Berendsen thermostat with a large time constant τ. In a production run, Berendsen τ must be large, compared to the characteristic time of the process of interest.
For MD runs where the initial configuration is already well equilibrated (eg taken from a previous MD run using an empirical forcefield), the equilibration should start with the `LOCAL` thermostat, and omit `SCALING` altogether.

4.7.6 Keyword CONSERVE

The keywords requests imposition of constraints on some or all mechanical constants of the overall motion of the system.
Options:
NONE / **POSIT**ION / **MOMEN**TUM / **ANGUL**AR / **ALL**

NONE	Do not impose constraints. This is the default, except for `MDYNAMICS RANDOM`.
POSITION	Reset position of the centre of mass of the system to (0,0,0) on each MD step.
MOMENTUM	Reset overall momentum of motiom of the system to zero on each MD step.
ANGULAR	Reset overall angular momentum of the system to zero on each MD step. This constraint makes no sense, and is ignored, for periodic calculations.
ALL	Equivalent to `POSITION MOMENTUM ANGULAR`. This is the default for `MDYNAMICS RANDOM`

Description:
For infinitely accurate arithmetics, and an infinitely small time step, overall constants of motion must be conserved automatically. Unfortunately, in the presence of numerical noise and residual errors in the time evolution algorithms, mechanical constants of motion

are no longer conserved in practice. For example, it is easy to show that the residual error of the velocity Verlet algorithm (used in deMon) with lead to an unbound amplification of the angular momentum of a diatomic molecule.
Additionally, in many cases, we are interested in the time evolution of a system which is at rest with respect to the laboratory frame. However, constructing an initial guess with zero initial momenta may be cumbersome.
Therefore, the `CONSERVE` keyword provides a band-aid for both problems. deMon keeps track of the total energy loss due to these constraints.

4.7.7 Keyword MDCONSTRAINTS

This keyword imposes atomic constraints during a molecular dynamics simulation.
Description:
Atomic positions or separate X, Y, Z Cartesian coordinates of atoms can be constrained, using the following syntax:

	POSITION atom1	XYZ coordinates of atom1 are fixed
or	POSITION atom1, atom3	XYZ coordinates of atom1 and atom3 are fixed
or	POSITION atom1-atom3	XYZ coordinates of atom1 through atom3 are fixed
or	Γ atom1, atom3	Γ, i.e. X or Y or Z coordinate of atom1 and atom3 fixed
or	Γ atom1-atom3	Γ, i.e. X or Y or Z coordinate of atom1 through atom3 are fixed

The default is no constraints.
Other constraint types will be added in the future.

4.7.8 Keyword MDPRESSURE

Activates constant-pressure molecular dynamics. Options:

P=	External pressure in MPa. The default is 0.1
TAU=	Berendsen Coupling constant in ps. The default is 1.0.
COMP=	Compressibility of the system in 1/MPa. The default is set to be reasonable for water.

Description:
This keyword requires periodic boundary conditions to be active (see section 4.9.6). Be-

cause periodic boundary conditions are currently implemented only for MM and QM/MM runs (see section 4.5.1), constant-pressure dynamics necessarily implies either MM or QM/MM.

With `MDPRESSURE` on, deMon will adjust the size of the periodic box to maintain the desired pressure. The periodic box parameters are controlled as described for the Berendsen thermostat (see eq. 13 in section 4.7.5). The corresponding time constant is defined by the `TAU=` parameter. Practical implementation of the Berendsen pressure bath also requires an estmation of the compressibility of the system, which is specified with the `COMP=` keyword. This value need not to be exact, as it enters only the calculation of the coordinare scaling factor for coordinates.

Compressibilities of some representative systems are:

System	Compressibility, MPa^{-1}
gases at 0.1 Mpa	≈ 10
water	$\approx 500 \times 10^{-6}$
solids	$\approx$ 1-20 $\times 10^{-6}$

Warning: Constand-pressure molecular dynamics only makes sense if the system is sufficiently large (thousands of particles) and/or time scales are reasonably long (tens of picoseconds).

4.8 Property Control

4.8.1 Keyword POPULATION

This keyword requests population analysis. In the absence of the POPULATION keyword, the default is to skip population analysis.
Options:
MULLIKEN / LOEWDIN / BADER

MULLIKEN	A Mulliken population analysis is performed. This is the default.
LOEWDIN	A Löwdin population analysis is performed.
BADER	A Bader population analysis is performed (unimplemented).

Description:
In all three cases, deMon calculates atomic charges and, in the case of open-shell systems, atomic spin charges. For the Mulliken [92] and Löwdin [93] population analysis, the bond order [94] or valence matrix [95] are calculated as well (closed-shell systems only). In the case of the population analysis of Bader [96] critical points of the electronic density and the molecular graph are calculated. These quantities can be visualized with Vu (see 8) using the `deMon.pie` file.

4.8.2 Keyword DIPOLE

This keyword activates the calculation of molecular electrostatic moments.
Description:
With the DIPOLE keyword the calculation of the electrostatic dipole, quadrupole and octupole moment is activated. If higher moments are required the parameter MAXMOM in the `parameter.h` file has to be adjusted. The electrostatic moments are calculated from the orbital density (see Section 1.4) independent from the chosen energy approximation. If the electrostatic moments should be calculated in STANDARD ORIENTATION (see Section 4.1.4 for the definition) a single point calculation with the symmetry keyword activated (SYMMETRY ON) has to be performed.

4.8.3 Keyword MAGOUT

This keyword requests writing of the interface files for the NMR/EPR program MAG.
Options:

MASTER / **MAG**

MASTER	Produce output for MASTER NMR programs (see section **??**).
MAG	Produce output for MAG NMR program (see section 9).

Description:

If `MAGOUT MASTER` is given, four interface files, `deMon.nmr11`, `deMon.nmr70`, `deMon.nmr71` and `deMon.nmr72` are written. These are data files for the MASTER program (see Section **??**. The file `deMon.nmr11` contains the grid information (points and weights). The other files, `deMon.nmr70`, `deMon.nmr71` and `deMon.nmr72` contain dimension, pointer and molecular (coordinates, basis, etc.) informations, respectively. In the case of an EPR calculation the file `deMon.nmr11` is not produced. It should be noted that for NMR and EPR calculations the angular momentum for the auxiliary functions is limited to $l = 2$.
If `MAGOUT MAG` is given, deMon will create two interface files, `deMon.nmr11` and `deMon.nmr50`, in the format expected by the MAG program (see Section 9). Note that the version of the MAG program, included in this distribution, lacks some of the key features due to licensing restrictions on it's redistribution. An up-to-date version of the MAG code should be requested directly from Vladimir Malkin.
The use of the MASTER and MAG programs is described in Sections **??** and 9.

4.8.4 Keyword **POLAR**IZABILITY

This keyword activates calculation of the polarizabilities and hyperpolarizabilities.
Options:

ALPHA / **BETA** / **GAMMA**

ALPHA	The polarizability tensor $\boldsymbol{\alpha}$ is calculated. This is the default.
BETA	The first hyperpolarizability β is calculated.
GAMMA	The second hyperpolarizability γ is calculated.
FFS=<Real>	Finite field strength used in the polarizability calculation.
EFISH	Activate the EFISH orientation.

Description:

The polarizabilities are calculated by the finite field method [97] using field strengths of 0.003 a.u. for α and β and 0.01 a.u. for γ. The field value can be modified by the option FFS. The hyperpolarizabilities β and γ are always calculated in the so-called EFISH (Electric Field Induced Second Harmonic generation) orientation. In this orientation the

z axis of the molecule is oriented along the permanent dipole moment of the system. The mean first and second hyperpolarizabilities are defined in EFISH orientation as:

$$\overline{\beta} = \frac{3}{5}\sum_i \beta_{iiz} \tag{4.14}$$

$$\overline{\gamma} = \frac{1}{5}\sum_{i,j} \gamma_{iijj} \tag{4.15}$$

For the calculation of the polarizability tensor $\boldsymbol{\alpha}$ no special orientation is used. With the option EFISH the EFISH orientation can be enforced. The mean polarizability $\overline{\alpha}$ and polarizability anisotropy $|\Delta\alpha|^2$ are calculate as:

$$\overline{\alpha} = \frac{1}{3}\left(\alpha_{xx} + \alpha_{yy} + \alpha_{zz}\right) \tag{4.16}$$

$$|\Delta\alpha|^2 = \frac{3\mathrm{tr}\,\boldsymbol{\alpha}^2 - (\mathrm{tr}\,\boldsymbol{\alpha})^2}{2} \tag{4.17}$$

Requesting β or higher polarizabilities changes default for the grid generating function to `GUESS` (See section 4.4.9).

4.8.5 Keyword **FREQU**ENCY

This keyword activates frequency analysis.
Options:

RAMAN	The Raman intensities are calculated.
RESTART	The frequency analysis is restarted from the `deMon.rst` file.
VIB=<Real>	Scaling factor for the numerical step size.

Description:
By default only the infra-red adsorption intensities of the harmonic normal modes are calculated, using the dipolar approximation. With the option RAMAN, the Raman intensities (in atomic units) and the depolarization ratio [98] are calculated, too. This is considerably more time consuming conpared to the standard frequency analysis. The RESTART option permits the restart of a frequency analysis. The Hessian matrix elements already calculated are read from the restart file `deMon.rst` and the analysis is continued. The VIB option allows adjustments to the numerical step size, used for differentiation of the ananytical gradients. By default, step size of 0.001 Bohr is used for `VXCTYP BASIS` (see section 4.4.8). If `VXCTYP AUXIS` is used, the default displacement is changed to 0.025 Bohr. For examples, with VIB=2 (and `VXCTYP BASIS`) the step size becomes 0.01 Bohr.

4.8.6 Keyword THERMO

This keyword activates the calculation of thermodynamic functions. The THERMO keyword can only be used in combination with the FREQUENCY keyword (see Section 4.8.5).
Options:

MAX=<Real>	Maximum temperature. The default is 2000 K.
MIN=<Real>	Minimum temperature. The default is 100 K.
INT=<Real>	Temperature interval. The default is 100K.

Description:
The thermodynamic functions are calculated in the approximation of the ideal polyatomic gas [99] and harmonic vibrational approximation. For diatomic molecules a ro-vibronic correction term is included (see code). The electronic contributions are neglected, apart from the trivial contribution due to the multiplicity of the ground electronic state. The options MAX and MIN allow specification of the temperature range. The step size of the temperature interval is defined by the option INT. The output contains the heat capacities c_p, the entropy S, the enthalpy H and the inner energy U for each temperature. Also the natural logarithm of the partition function $z = z_{trans} z_{rot} z_{vib} z_{el}$ is given. Imaginary normal modes are not included in the summation.

4.8.7 Keyword FNMC

Requests finite nuclear mass correction. Options:
OFF / **ON**
Description:
With this keyword, the finite nuclear mass correction to the self-atomic elements of the kinetic energy matrix are switched on. Before you use this you should read the literature, e.g. ref. [100].

4.8.8 Keyword HARDNESS

Compute orbital-dependent reactivity indices hardness, softness and Fukui function.
Options:

DELN= Energy perturbation (in electrons). The defalt is 0.01

FPMO= First perturbed molecular orbital. The default is 1

TOL= Tolerance for degeneracy detection (in Hartree). The default is 10^{-5}

Description:

Compute orbital-dependent reactivity indices hardness, softness and Fukui function, as given in a supplied preprint by Mineva and Heine.

4.9 Visualization and Topology

The keywords PLOT, ISOSURFACE and GEOSURFACE of this section are exclusive and cannot be combined. The same holds for the keywords BOX and POINTS.

4.9.1 Keyword VISUALIZATION

This keyword activates the interface to the visualization programs MOLDEN [101] and MOLEKEL [102]. Options:
MOLDEN / **MOLEK**EL

MOLDEN	The MOLDEN interface is activated. The MOLDEN input is written to the file `deMon.mol`. This is the default.
MOLEKEL	The MOLEKEL interface is activated. The MOLEKEL input is written to the file `deMon.mkl`.

XYZ / **OPT** / **FULL**

XYZ	Only the molecular structures and energies of each optimization step are written to the `deMon.mol` file. This is the default if the option MOLDEN or MOLEKEL is missing.
OPT	The molecular structures as well as the forces and step sizes of each optimization step are written to the `deMon.mol` file. This option triggers the MOLDEN interface.
FULL	A full MOLDEN or MOLEKEL input is written. This is the default if the option MOLDEN or MOLEKEL is set.

Description:
For large systems the FULL option of the VISUALIZATION keyword produces very large MOLDEN and MOLEKEL input files. Therefore, it is recommended to use the VU interface for the plotting of molecular fields of large systems (see the keyword PLOT and ISOSURFACE below). In the case of a frequency analysis the FULL option is activated by default in order to animate the vibrations.
For molecular dynamics runs, only `VISUALIZATION MOLDEN XYZ` is supported. This option will be used, regardless of the parameters supplied here.

4.9.2 Keyword PLOT

This keyword controls the calculation and plotting of molecular fields. See the keywords BOX (4.9.6) and POINTS (4.9.7) for the definition of the plot support.
Options:

⟨**FIELD**⟩	Molecular field specification. The available field acronyms are given in Table 9. This option is mandatory!

BASIS / **AUXIS**

BASIS	The orbital density is used for the construction of the plot function. This is the default.
AUXIS	The auxiliary function density is used for the construction of the plot function. This option is incompatible with the READ option.

BINARY / **ASCII** / **TABLE**

BINARY	A binary file `FIELD.bin` is written containing the coordinates and plot function values. The VU file format is used. The VU control file `deMon.pie` is written, too. This is the default.
ASCII	An ascii file `FIELD.asc` is written containing the coordinates and plot function values.
TABLE	A function table is written in the output file `deMon.out`.
READ	Specifies that an orbital list is read in the keyword body of PLOT. This keyword is incompatible with the EFIDS field.

Description:
The definition of the molecular field is mandatory for the PLOT keyword. The available molecular fields are listed in Table 9. If the BOX and POINTS keywords are missing the default BOX setting is used for the plot. Only one molecular field at a time can be calculated. If several fields have to be calculated restart calculations with SCFTYPE MAX=0 and GUESS RESTART (see 4.4.1 and 4.4.4) may be performed. In any case, no more than 20 molecular field entries at a time can be calculated. A field with its first and second derivatives counts for 10 entries (1 + 3 + 6).

The AUXIS and BASIS option specify the density (auxiliary function or orbital density, respectively) used for the calculation of the molecular field. The AUXIS option is incompatible with the READ option. If the auxiliary function density is used for the calculation of the molecular field a considerable speed up will be achieved. However, the field values may show deviations with up to 10% compared to the orbital density. Nevertheless, the field topology is in most cases qualitatively correct.
By default, the PLOT keyword produces two additional output files, the binary file `FIELD.bin` and the VU control file `deMon.pie`. With these files a VU session can be started (see Section 8). With the ASCII option only the file `FIELD.asc` is created. The coordinates and molecular field values are listed in this file. With the TABLE option a function table is created in the output file.
With the READ option a set of molecular orbitals for the calculation of the molecular field can be selected. This option cannot be used for the EFIDS field. In the case of the PSI and D1PSI field the READ option specifies directly the calculated molecular orbitals. Again, not more than 20 orbitals at a time can be calculated (in the case of D1PSI not more than 5). If for the other fields orbitals are selected than the field is constructed from the sum of the selected orbitals. E.g. the following input specifies the calculation of the density using the molecular orbitals 1, 2 and 4:

```
PLOT RHO READ
1 2 4
```

Virtual orbitals can be included. They will be occupied by one electron. In the case of an unrestricted calculations (UKS option of keyword SCFTYPE; see 4.4.1) α and β-orbitals can be selected, each set in a separate input line. In the following example the α-orbitals 1, 2, 3, 4, 5, 6, 7 and the β-orbitals 1, 2, 3, 4, 5, 6 are selected for the calculation of the molecular electrostatic potential:

```
PLOT ESP READ
1 2 3 4 5 6 7
1 2 3 4 5 6
```

If only the β-orbitals should be selected the first line in the above example must have a 0 entry:

```
PLOT ESP READ
0
1 2 3 4 5 6
```

Table 9: Molecular fields available in deMon

Acronym	Description
PSI	Molecular orbitals. By default the five highest occupied and lowest unoccupied orbitals are calculated.
D1PSI	Molecular orbitals and first derivatives. By default the HOMO and LUMO are calculated
RHO	Electron density. By default the density of the occupied orbitals is calculated.
D1RHO	Electron density and first electronic derivatives. By default the density and its derivatives of the occupied molecular orbitals are calculated.
D2RHO	Electron density, first and second electronic derivatives. By default the density and its derivatives of the occupied orbitals are calculated.
SPIN	Spin density. By default the spin density of the occupied molecular orbitals is calculated.
D1SPIN	Spin density and first electronic derivatives. By default the spin density and its derivatives of the occupied molecular orbitals are calculated.
LAP	Electron density Laplacian. By default the Laplacian of the occupied molecular orbitals is calculated.
ESP	Molecular electrostatic potential. By default the MEP of the occupied molecular orbitals is calculated. A minus sign in front indicates that only the electronic part is calculated.
D1ESP	Molecular electrostatic potential and its first derivatives. By default the MEP and its derivatives of the occupied orbitals are calculated.
D2ESP	Molecular electrostatic potential and its first and second derivatives. By default the MEP and its derivatives of the occupied orbitals are calculated.
EFI	Molecular electric field. By default the electric field of the occupied molecular orbitals is calculated.
D1EFI	Molecular electric field and its first derivatives. By default the electric field and its derivatives of the occupied orbitals are calculated
EFG	Molecular electric field gradients. Identical to D1EFI.
EFIDS	Anion surface generation [103].
ELF	Electron localization function [104,105].

4.9.3 Keyword **CPSEA**RCH

This keyword activates the critical point search of scalar molecular fields. For the definition of the search area see the keywords BOX (4.9.6) and POINTS (4.9.7).
Options:

⟨**FIELD**⟩ Scalar molecular field specification. The available field acronyms are RHO and ESP. See Table 9 for the acronym meanings. This option is mandatory!

BASIS / **AUXIS**

BASIS The orbital density is used for the calculation of the scalar molecular field. This is the default.

AUXIS The auxiliary function density is used for the calculation of the scalar molecular field.

Description:
The critical points of the density (RHO) and molecular electrostatic potential (ESP) can be searched with this option. By default, the critical point search is performed in a marching cube like style [106], embedding the molecule in a rectangular box (BOX LARGE; see Section 4.9.6), which is in turn divided into subboxes. Each subbox is then searched for a critical point [107]. The box size and shape can be manipulated by the BOX keyword. As an alternative, start points for the critical point search can be supplied by the POINTS keyword (see Section 4.9.7). With the POINTS option POLYGON start points between atom pairs and triples are automatically generated. This option is recommended for the critical point search of the density. In addition to the critical point search the molecular connectivity over (3,-1) critical points is generated [96].
`EFFECT OF AUXIS ON THE CPS -> SIGFRIDO`

4.9.4 Keyword **ISOSU**RFACE

This keyword controls the calculation and plotting of molecular field isosurfaces. Only one isosurface at a time can be generated. For the generation of multiple isosurfaces the restart description of the keyword PLOT (Section 4.9.2) can be applied. See the keyword BOX (4.9.6) for the definition of the isosurface boundary.
Options:

⟨**FIELD**⟩ Molecular field specification. The available field acronyms are PSI, RHO, SPIN, LAP, ESP, EFIDS and ELF. See Table 9 for the acronym meanings. This option is mandatory!

BASIS / **AUXIS**

BASIS	The orbital density is used for the construction of the isosurface. This is the default.
AUXIS	The auxiliary function density is used for the construction of the isosurface. This option is incompatible with the READ option.

LINEAR / **BILIN**EAR / **LOGAR**ITHMIC

LINEAR	Linear interpolation scheme for the isosurface construction. This is the default.
LOGARITHMIC	Logarithmic interpolation scheme for the isosurface construction

BINARY / **ASCII** / **TABLE**

BINARY	A binary output of the isosurface is written in the file `LAT.bin` using the VU file format. The VU control file `deMon.pie` is written, too. This is the default.
ASCII	An ascii output of the isosurface is written in the file `deMon.lat`.
TABLE	A function table of isosurface coordinates is written in the output file `deMon.out`.

READ	Specifies that an orbital list is read in the body of ISOSURFACE.
ISO=<Real>	Isosurface value. This value is mandatory!
TOL=<Real>	Tolerance for data reduction.

Description:
The options AUXIS, BASIS and READ of the ISOSURFACE keyword are identical to the corresponding options of the PLOT keyword (4.9.2). The BINARY option specifies that the isosurface coordinates and connectivities are written into the binary file `LAT.bin`. With the VU control file `deMon.pie`, that is also generated, the isosurface grid can be visualized in VU (see 8). It is important to note that the `LAT.bin` file can be used as an input file for the plotting of a molecular field on the isosurface! For this purpose this file has to be in the same directory as the `deMon.inp` file and the POINTS keyword (see Section 4.9.7) has to be used to define the isosurface in `LAT.bin` as plot support. With the ASCII option the ascii file `deMon.lat` is generated. This file contains the following data:

```
RHO ( .100000E+00 A.U. ) ISOSURFACE COORDINATES IN ANGSTROM
```

```
NUMBER OF VERTICES: 1814
NUMBER OF FACETS: 3624
VOLUME: .163076E+04
AREA: .176664E+03

        X               Y               Z
    .312196E+00    -.729474E+00     .105260E+01
    .243158E+00    -.765796E+00     .105260E+01
    .243158E+00    -.729474E+00     .917119E+00
       ...             ...             ...
CONNECTIVITY

          1          2          3
          4          5          6
          6          7          8
         ...        ...        ...
```

The first line is the file header including the field information (RHO), the isovalue (0.1 a.u.) and the units (ANGSTROM) used for the coordinates, area and volume data. The NUMBER OF VERTICES correspond to the number of interpolation points for the isosurface. The NUMBER OF FACETS is the number of surface elements of the isosurface. The volume (here in $Å^3$) and the surface area (here in $Å^2$) are then given. It should be noted that the volume includes all volume elements with field values under the given threshold. Therefore, the volume of an 0.1 a.u. isosurface of a field with positive and negative values includes all volume elements with positive field values less than 0.1 a.u. **and** all volume elements with negative field values. The next block represents the coordinates of the interpolation points (here in Å). The last block, entitled CONNECTIVITY, describes the triangulation of the interpolation points. With the option TABLE the same data as with the ASCII option are written into the file `deMon.out`.

The options LINEAR and LOGARITHMIC specify the interpolation scheme for the construction of the isosurface using the marching tetrahedron algorithm [106]. The value of the isosurface is specified with the ISO option. A value in vertical delimiters (e.g. |0.1|) indicates an absolute isosurface value. It can be used to generated simultaneously two isosurfaces with the same positive and negative threshold (e.g. for molecular orbitals). If the threshold of the two isosurfaces should be different (e.g. for molecular electrostatic potentials) the two values can be specified as:

```
ISO = 0.1/-0.05
```

The ISO option has no default setting and, therefore, is mandatory! With the TOL

option the amount of data reduction is specified. Allowed values range from 10^{-3} to 0.5, specifying the severity of the data reduction. A value of 10^{-3} indicates no data reduction at all. This is the default. With increasing TOL values the data reduction increases, thus decreasing the memory request and at the same time the resolution.

4.9.5 Keyword GEOSURFACE

This keyword controls the calculation and plotting of geometrical surfaces like spheres or ellipsoids. See the keyword BOX (4.9.6) for the definition of the geosurface boundary.
Options:
SPHERE / ELLIPSOID

SPHERE	A sphere with the radius a around the origin $\vec{r}_o$ is constructed.
ELLIPSOID	An ellipsoid with the semi-axes a, b and c around the origin $\vec{r}_o$ is constructed.

BINARY / ASCII / TABLE

BINARY	A binary output of the geometrical surface is written in the file `LAT.bin` using the VU file format. The VU control file `deMon.pie` is written, too. This is the default.
ASCII	An ascii output of the geometrical surface is written in the file `deMon.lat`.
TABLE	A function table of the geometrical surface coordinates is written in the output file `deMon.out`.

TOL=<Real>	Tolerance for data reduction.

Description:
The options BINARY, ASCII, TABLE and TOL are identical to the corresponding options of the ISOSURFACE keyword. The option SPHERE or ELLIPSOID is mandatory for the GEOSURFACE keyword. The origin and radius or semi-axes of the sphere or ellipsoid have to be defined in the keyword body of GEOSURFACE. For a sphere with the origin at $\vec{r}_o = (1, 3, 4)$ and a radius of 7 the input has the form:

```
GEOSURFACE SPHERE
```

```
1.0 3.0 4.0
7.0
```

The units are defined by the unit definition in the GEOMETRY keyword (see Section 4.1.1). For an ellipsoid the following input form is used:

```
GEOSURFACE ELLIPSOID
1.0 3.0 4.0
7.0 1.0 0.5
```

The definition of all three semi-axes is mandatory.

4.9.6 Keyword BOX

With this keyword the box around a molecule for plotting (4.9.2), critical point search (4.9.3) and iso- and geosurface generation (4.9.4 and 4.9.5) is defined.
Options:
STANDARD / **SMALL** / **LARGE**

STANDARD	A box enclosing all defined atoms with an extension of two times there covalent radii. This is the default.
SMALL	A box enclosing all defined atoms with an extension of 1.2 times there covalent radii.
LARGE	A box enclosing all defined atoms with an extension of four times there covalent radii. This is the default for the critical point search.

MEDIUM / **COARS**E / **FINE**

MEDIUM	The mesh spacing of the box is 0.3 Bohr. In the case of the critical point search it is 0.5 Bohr. This is the default.
COARSE	The mesh spacing of the box is 0.9 Bohr. In the case of the critical point search it is 1.0 Bohr.
FINE	The mesh spacing of the box is 0.1 Bohr.
READ	The box definition is read from the keyword body.

Description:
With the options STANDARD, SMALL and LARGE the size of the box is specified. The box will be every time created in STANDARD ORIENTATION (see 4.1.4) and, therefore, is symmetry adapted to the system. The options MEDIUM, COARSE and FINE define

the mesh spacing within the box. The FINE mesh spacing will generate huge data files and, thus, is only applicable for small systems. With the READ option the box can be explicitly defined in the keyword body of BOX. If only a part of the molecule should be considered the box can be defined simply by the atoms that should be included. In this case the STANDARD, SMALL and LARGE, as well as the MEDIUM, COARSE and FINE options remain active. For a SMALL box with FINE resolution containing the H1 and O atom of the water molecule the input has the form:

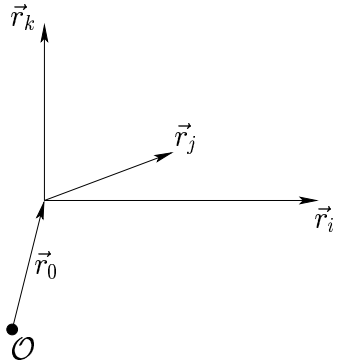
```
BOX SMALL FINE READ
O H1
```

Please note that the atoms are all defined in one input line. Instead of the explicit definition of each atom, elements can be used for the box definition, too. Therefore a SMALL box with MEDIUM resolution containing only the hydrogen atoms of water is defined as:

```
BOX SMALL MEDIUM READ
H
```

Alternatively, the plot box can be defined by vectors in the keyword body of BOX. In this case the options STANDARD, SMALL and LARGE, as well as MEDIUM, COARSE and FINE loose their meaning. The following format is used for the box definition with vectors:

```
BOX READ
NI  NJ  NK
XO  YO  ZO
XI  YI  ZI
XJ  YJ  ZJ
XK  YK  ZK
```



where `NI`, `NJ` and `NK` specify the number of points in the corresponding $\vec{\imath}$, $\vec{\jmath}$ and $\vec{k}$ directions. The $\vec{r}_0 = (x_0, y_0, z_0)$ position vector defines the origin of the box. The other three orthogonal vectors $\vec{r}_i = (x_i, y_i, z_i)$, $\vec{r}_j = (x_j, y_j, z_j)$ and $\vec{r}_k = (x_k, y_k, z_k)$ span the box as sketched above. In the case of a plane (two-dimensional box) the definition of `NK`, `XK`, `YK` and `ZK` are obsolete. For a line (one-dimensional box) only `NI`, $\vec{r}_0$ and $\vec{r}_i$ have to be defined. The explicit definition of a plot box is described in Example 5.17.

4.9.7 Keyword POINTS

With this keyword points for the plotting of molecular fields (4.9.2) and start points for the critical point search (4.9.3) can be specified.
Options:
BINARY / **ASCII** / **POLYG**ON / **READ**

BINARY	The plot point coordinates and connectivities are read from the external binary file `LAT.bin`. This is the default.
ASCII	The plot point coordinates and connectivities are read from the external ascii file `LAT.asc`.
POLYGON	Starting points for the critical point search are generated by the polygon algorithm.
READ	The plot point coordinates are read from the input file `deMon.inp`.

Description:
With the option BINARY coordinates and connectivities are read from the file `LAT.bin`. Usually this file has been produced by a previous run calculating isosurfaces or geometrical surfaces (see Sections 4.9.4 and 4.9.5). In this way molecular fields can plotted on isosurfaces. Example 5.17 describes the plotting of the electrostatic potential on the isodensity surface of benzene. The `LAT.bin` file and the corresponding plot output can be visualized with VU (see 8). With the ASCII option plot point coordinates, and optional connectivities, can be read from the external ascii file `LAT.asc`. The file format is:

```
1814
    .312196E+00    -.729474E+00     .105260E+01
    .243158E+00    -.765796E+00     .105260E+01
    .243158E+00    -.729474E+00     .917119E+00
       ...             ...             ...
         1              2              3
         4              5              6
         6              7              8
        ...            ...            ...
```

The first integer number (here 1814) denotes the number of points. For each point one line with its coordinates (x, y and z) is then given. The integer triples after the coordinate specification are the connectivities. This part of the input is optional. The `LAT.asc` file can be easily generated from the `deMon.lat` file (see 4.9.4 and can directly be used for the visualization with VU.

With the POLYGON option starting points for the critical point search are automatically generated. In the case of the electronic density this algorithm usually generates a sufficient set of starting points to find all critical points. **This is not the case for the critical points of other molecular fields**. The READ option can be used to read point coordinates from the input file `deMon.inp`. The coordinates are given in free format in the keyword body of POINTS, one input line for each point.

4.10 Miscellaneous Keywords

4.10.1 Keyword MAXMEM

This keyword specifies the amount of memory available to the job.
Options:

<integer>	Amount of memory available. The default is 256 Mbytes.
KBYTE	Units: Kilobytes.
MBYTE	Units: Megabytes. This is the default.
GBYTE	Units: Gigabytes.

Description:
deMon can use the `MAXMEM` value for choosing the appropriate handling of `ERIS MEMORY` (see 4.4.3) and the value of `MATDIA THRESHOLD=` (see 4.10.6).
The amount of memory specified in `MAXMEM` cannot exceed the `MAXRAM` compilation-time parameter.

4.10.2 Keyword TITLE

This keyword specifies the job title.
Options: None
The job title is limited to 60 characters and the title line cannot be continued over more than one line.

4.10.3 Keyword PRINT

This keyword controls optional printing.
Options:

ALWAYS	Print requested output for all calculation phases.
ATOMMAP	Print atom ordering maps.
AUXIS	Print auxiliary function table.
BASIS	Print molecular orbitals in long format.
CGTO	Print GTO contraction table.
COORD	Print primitive internal coordinates.

DE2	Print Hessian matrix.
DEBUG	Generate debug output.
ECP	Print ECP table.
EMBED	Print embedding charges.
ERIS	Print three-center electron repulsion integrals.
G	Print auxiliary function Coulomb matrix.
GRID	Generate full grid output.
GTO	Print primitive GTO table.
KS	Print Kohn-Sham and core Hamiltonian matrix.
MAX=<Integer>	Maximum number of SCF cycles with print output.
MD	Print detailed MD output.
MDEXTRA	Print insanely detailed MD output .
MM	Print additional MM output.
MOE	Print molecular orbital energies and occupations.
MOS	Print molecular orbital energies, occupations and coefficients.
OPT	Generate full optimization output.
ORTHO	Print orthogonalization matrix $\boldsymbol{S}^{-1/2}$.
P	Print density matrix.
POPAN	Generate full population analysis output.
PBC	Print periodic bondary conditions information.
QMMM	Print additional QM/MM information.
RAM	Print RAM allocation table.
S	Print overlap matrix.
SYMMETRY	Generate full symmetry output.
T	Print kinetic energy matrix.
TB	Print tight-binding matrices.
XCE	Generate full exchange-correlation energy output.
XCV	Generate full exchange-correlation potential output.

Description:
With the MAX option the number of SCF cycles with active print output can be specified. All matrices that are changing during the SCF (e.g. P, KS, MOS, etc.) can be printed for each cycle. The following example activates the printing of the density matrix in the first 10 SCF cycles:

```
PRINT MAX=10 P
```

Printing of molecular orbitals can be limited with the MOS option. The input `MOS = <First> - <Last>` specifies that only the MOs in the range from `<First>` to `<Last>` are printed. Here `<First>` and `<Last>` refer to the (integer) numbers of the molecular orbitals. With the MOE option the printing of the molecular orbital coefficients can be suppressed. In order to print the orbital energies and occupation numbers of the orbitals 10 to 25 the following input can be used:

```
PRINT MOS=10-25 MOE
```

Without MOE the MO coefficients would be printed, too.
With the DEBUG option a complete printing of all matrices is activated! Of course such a output is huge and should only be used for debugging.

4.10.4 Keyword EMBED

This keyword specifies the embedding into point charges.
Options:
FILE / READ

FILE	The embedding point coordinates and charges are read from the deMon.cub file. This is the default.
READ	The embedding point coordinates and charges are read from the input file.

Description:
The coordinates and charges are given in free format as real numbers either in the embedding file deMon.cub (see Table 3 or in the keyword body of EMBED. The Cartesian coordinates of the embedding points refer to the input orientation of the molecule as defined in the input file. If a Z-Matrix input is used the embedding point coordinates are defined with respect to the Z-Matrix input orientation (see Figure 1). The point charge coordinate unit (Ångström or atomic units) correspond to the unit system selected for the input coordinates (see keyword GEOMETRY in 4.1.1). Optional, an element symbol or a radius (real number) may be specified after the point charge coordinates and the charge values. This information will be used for the bond drawing in a Vu output of the embedding region (see Section 8). The following input specifies an embedding in a (distorted) octahedron of positive point charges:

```
EMBED READ
 5.0  0.0  0.0  1.0  H
-5.0  0.0  0.0  1.0  H
 0.0  5.0  0.0  1.0  H
 0.0 -5.0  0.0  1.0  H
 0.0  0.0  2.0  1.0  1.5
 0.0  0.0 -2.0  1.0  1.5
```

The first line is only necessary if the embedding points are defined within the input file. In this example, a radius of the hydrogen atom is assigned to the first four embedding points. For the two others the radius is explicitly defined to 1.5 Ångström or atomic units depending on the unit system specification in the GEOMETRY keyword (see Section 4.1.1).

4.10.5 Keyword CHOLESKY

With this keyword a Cholesky decomposition for the inversion of the auxiliary Coulomb matrix can be selected.

Options:

OFF / **ON**

OFF	Singular value decomposition is used. This is the default.
ON	Cholesky decomposition is used.
SVDTOL=	Tolerance for singular value decomposition. Default is 10^{-5}.
SVDTO2=	

Description:

The Cholesky decomposition of the auxiliary function Coulomb matrix is considerably faster than the singular value decomposition (SVD). However, it may introduce numerical instabilities for large auxiliary functions sets. This may result in convergence failures of the SCF procedure. The Cholesky and singular value decomposition results are identical if the SVD nullspace is empty. By default, the SVD procedure will eliminate eigenvalues of the auxiliary Coulomb matrix smaller than 10^{-5}. This cut-off is appropriate in the vast majority of cases. However, if necessary, it can be modified with the `SVDTOL=` parameter. Specifying `SVDTO2=` will activate gradual quenching of the eigenvalues (see `svdmat.f`).

4.10.6 Keyword MATDIA

This keyword chooses matrix diagonalization technique.

Options:
RS / **DSYEV** / **D&C** / **DSYEVD** / **DSYEVR** / **JACOBI**

RS	EISPACK Householder.
DSYEV	LAPACK block Householder. Default for "large" matrices.
D&C	This is another name for `DSYEVD`
DSYEVD	LAPACK divide and conquer. Default for "small" matrices.
DSYEVR	LAPACK Relatively Robust Representations diagonalization.
JACOBI	Jacobi diagonalization. This is the default for atoms.
SMALL=	Diagonalizer to use for "small" matrices.
LARGE=	Diagonalizer to use for "large" matrices.
THRESHOLD=	Maximum side dimension of a "small' matrix.

Description:
The fastest diagonalizer available in deMon in the divide and conquer diagonalization from LAPACK [108]. However, it might fail in special situations. Fail safe diagonalizers in deMon are based on the Householder algorithm. For large systems, the DSYEV diagonalizer is considerably faster than the RS diagonalizer. However, it is also more memory demanding. Therefore, it should only be used if enough RAM space is available. The JACOBI diagonalizer is much slower but produces very pure eigenvectors. It is used by default for atomic calculations.
Normally, deMon will make the optimal diagonalizer choice based on the amount of memory available (see `MAXMEM`, section 4.10.1).

4.10.7 Keyword **WEIGH**TING

This keyword specifies the weight function used for the grid partitioning.
Options:
SCREENED / **BECKE**

SCREENED	A screened weight function is used. This is the default.
BECKE	The original Becke weight function is used.

Description:
The BECKE weight function [109] introduces a cubic scaling in the grid generation step [110]. By introducing a new piece wise defined weight function [65] near linear scaling

for the grid generation can be obtained. This weight function is specified by the option SCREENED.

4.10.8 Keyword **QUADR**ATURE

Numerical quadrature scheme can be modified using this keyword.
Options:
GAUSS / **EULER**

GAUSS	A Gauss-Chebyshev radial quadrature is selected. This is the default.
EULER	A Euler-MacLaurin radial quadrature is selected.
RANDOM	Random rotation of the Lebedev grids is activated.
REFERENCE	A reference grid is generated.

Description:
`TO BE WRITTEN`

4.10.9 Keyword **ECPIN**TEGRATION

This keyword chooses numerical integration scheme, used in calculation of the effective core potentials.
Options:
COARSE / **MEDIU**M / **FINE**

COARSE	Use 67 radial shells, (50,194)p fixed grid. Nominal accuracy 10^{-6}
MEDIUM	Use 99 radial shells, (75,302)p fixed grid. Nominal accuracy 10^{-8}
FINE	Use 131 radial shells, (200,1202) fixed grid. Nominal accuracy 10^{-10}

Description:
`TO BE WRITTEN`

4.10.10 Keyword **GENAO**

Generate atomic orbitals. **This is an experimental keyword. Do not use**.
Options:

CUTOFF= Cutoff energy in Hartree.

AO= indices of atoms, in a following list of integers, whose basis will be transformed to AO's.

NOT= indices of atoms, in a following list of integers, whose basis be excluded from the AO transformation.

Description:

Default: GENAO is switched off. If GENAO is switched on, the default is that all atoms are transformed. CUTOFF = 0.0 by default.

If GENAO is specified, the GTO basis will be transformed to an atomic basis. For the forthcoming SCF computation, not all AO's need to be considered and hence the matrix size may be reduced. The number of AO's which will be considered in the SCF depends on the cutoff energy: Orbitals with higher atomic energy will be left out. If the cutoff energy is higher than the highest atomic energy this is a unitary transformation and gives exactly the same result as the original GTO computation.

5 Examples

THIS SECTION NEEDS TO BE WRITTEN. VOLUNTEERS?

5.1 Example ps18

5.2 Examples amk5, amk6, amk7, and amk8

5.3 Examples amk29, amk30, and amk31

5.4 Example ps86

5.5 Example th6

5.6 Example amk17

5.7 Example ps84

5.8 Example amk19

5.9 Example amk20

5.10 Example ps15

5.11 Example ps6

5.12 Example amk24

5.13 Example amk26

5.14 Example amk33

5.15 Example amk35

5.16 Example ps88

5.17 Example ps90

6 Using QM/MM in deMon

TO BE WRITTEN

7 Using MD in deMon

TO BE WRITTEN

8 Vu

This chapter contains an introduction to the basic operating instructions of the Vu graphical interface which is part of the visualization program Vu. It constitutes a subset of all the available options in the graphical interface of Vu. This chapter shows only the most important features required to start using Vu. Prior to its use, it is highly recommended to read Chapter 4.9 of this guide. For a complete description of the structure and operation of Vu, please consult:

- The User Manual of the Configurable Scientific Visualization Program Vu. Which contains full description of architecture, capabilities and limitations of Vu.
- Description of the file format. Describes the structure of `.pie` files which are written by deMon and read by the Vu graphical interface.
- Frequently Asked Questions on the Utilization of Vu. Contains answers to many of the doubts involved with the use of Vu.
- *http://www3.sympatico.ca/chantal.pic/vu/eng/index.html*. This is a temporarily address.

8.1 What is Vu?

Vu is a configurable visualization software tool for the display and analysis of numerical solutions. It is used for exploring and interpreting results from simulation programs or experimental measurements. Based on a dictionary and a grammar common to researchers of various domains, it finds applications in fluid mechanics, civil engineering, applied mathematics, manufacturing, injection molding, combustion, structures, computational chemistry, etc. Vu is available on computers of various sizes, from laptops using Linux to virtual reality immersion environments (CAVE, ImmersaDesk, Reality Center, etc.), on all Unix platforms.

Vu uses three ingredients to construct an image: a support, an entity and a mode. The support is the place where the image resides: a plane, sphere, cylinder, geometric, etc.

The entity is what it is represented graphically: a set of vectors like velocity, a scalar field like electron density, etc. There are eight types of entities: mesh, graph, iso, vector, tensor among others.

The mode is the way the user wants to see the image: static, dynamic, injection or animation.

8.2 Vu and deMon

As a part of deMon, there are a set of routines that build the required files for visualization with Vu. This allows the users to display many scalar fields, critical points, etc. and select the way of visualize it. Before using Vu, the user has to run deMon with the proper visualization keywords in order to generate the two required files for the graphical interface:

`deMon.pie`: The control file for Vu. It contains all the definitions for construct the images.

`FIELD.bin`: This file is referred by the `deMon.pie` file. Contains the values of the scalar field selected for plotting.

Most supporting definitions for chemistry used by deMon reside in the `deMonMacros.vu` file which is included automatically at run time. So, `deMonMacros.vu` file should reside in the location where Vu stands.

8.3 Running Vu

To start Vu first export the display to your current window, then, on the command line type:

```
Vu
```

and press Enter key. The program starts without any file and then you can get your own `deMon.pie` file from the menu or type:

```
Vu deMon.pie
```

The program starts with the structure associated to the `.pie` file that you typed. There are other ways to read files. See Vu reference manual.

When Vu starts, three windows appear on the screen:

1. Menu Window. Used to build images.
2. Visualization Window (Empty at startup or displaying the molecular structure if a `.pie` file was selected) Used to visualize and modify images.
3. Message Window. Used by Vu to show information extracted of the files and error messages.

The user can adjust the size and position of these windows with the mouse.

Menu window. This window is composed of:

1. a menu bar (File, Zones, Info, Options, Stereo, Help),
2. a list of images proposed by Vu,

3. the lists of Supports, Entities and Windows (components of the image)
See figure **??**.
Through the File option of the menu bar the user can open a `.pie` file among others. Or can import files in other formats (Gaussian, PDB, etc) or save profiles, data and images.
The Images part of this Menu window lets the user select the image to be displayed in the visualization window. By default the Geometry image is active at startup. To select other image just click with the mouse on the box at the left of the image desired. An unlimited number of images can be displayed simultaneously on the Visualization window.
Each time the user selects an image on the Image section, the corresponding Support, Entity and Window associated with is shaded in the bottom section of this window.
By clicking with the mouse these shaded Supports, Entities and Windows, the user open new windows where their attributes can be changed to modify the way the image is displayed. See chapter 8 of the Vu Reference Manual.

9 MAG

The deMon program prepares the data needed to run the MASTER program [114,115] for NMR (shielding tensor) and EPR (Fermi term and anisotropic hyperfine tensor) properties. The deMon program provides the ground-state electronic structure of a system in the absence of an external magnetic field and then writes out information (atomic coordinates, basis set, grid, MO coefficients, orbital energies) to different files. The MASTER program, in a second step, calculates the magnetic properties. The NMR calculation is valid for closed shell and EPR for open shell systems. Recall that if a ROKS calculation has been performed, there will be no spin polarization possible and the Fermi term will then be zero, except if the unpaired electron is in an s-type orbital. The interface routine prepares 4 output files (deMon.nmr11, deMon.nmr70, deMon.nmr71, deMon.nmr72) for NMR and 3 output files (deMon.nmr70, deMon.nmr71, deMon.nmr72) for EPR.
The MASTER program has been successfully used for various applications: NMR screening constants for all magnetic nuclei for closed shell systems and hyperfine structure calculations for open-shells. The IGLO bases are recommended to obtain good NMR results [116]. The README files available in the MASTER program provide useful information to the users.

10 Quick Keyword Reference

TO BE WRITTEN. VOLUNTEERS?

11 Troubleshooting

NEED TO BE WRITTEN. VOLUNTEERS?

A Automatic Generation of Auxiliary Functions

With the auxiliary function definition GEN-An and GEN-An*, $n = 1$, 2 and 3, automatically generated auxiliary function sets are selected in deMon. The GEN-An sets consist of s, p and d Hermite Gaussian functions. The GEN-An* sets possess also f and g Hermite Gaussians. Because the auxiliary functions are used to fit the electronic density they are grouped in s, *spd* and *spdfg* sets. The exponents are shared within each of these sets [24, 25]. Therefore, the auxiliary function notation (3,2,2) describes 3 s sets with together 3 functions, 2 *spd* sets with together 20 functions and 2 *spdfg* sets with together 42 functions (see also 4.2.2). The range of exponents of all auxiliary functions is determined by the smallest, $\zeta_{\min}$, and largest, $\zeta_{\max}$, primitive Gaussian exponent of the chosen basis set. Therefore, the GEN-An and GEN-An* automatically generated auxiliary function sets are different for different basis sets. The number of exponents N (auxiliary function sets) is given by:

$$N = \mathrm{Int}\,\frac{\ln(\,\zeta_{\max}/\zeta_{\min}\,)}{\ln(6-n)} \tag{A.1}$$

Here n is 2, 3 or 4 according to the chosen GEN-An or GEN-An* set. The exponents are generated (almost; as it is explained below) even tempered and splitted into s, *spd* and, if a GEN-An* set is requested, *spdfg* sets, The tightest (largest) exponents are assigned to the s sets, followed by the *spd* and, if exist, *spdfg* sets. The basic exponent from which the generation starts is defined as:

$$\zeta_o = 2\,\zeta_{\min}\,(6-n)^{(N-1)} \tag{A.2}$$

From this exponent the two tightest s set exponents, ζ_1 and ζ_2 are generated according to the formulas:

$$\zeta_1 = \left(1 + \frac{n}{12-2n}\right)\zeta_o \tag{A.3}$$

$$\zeta_2 = \frac{\zeta_o}{6-n} \tag{A.4}$$

The other s set exponents are generated according to the even tempered progression:

$$\zeta_{i+1} = \frac{\zeta_i}{6-n} \tag{A.5}$$

The ζ_o exponent of the following *spd* sets is also generated according to the progression (A.5). Based on this ζ_o exponent the exponents of the first two *spd* sets are calculated

with the formulas (A.3) and (A.4). The following *spd* set exponents are then calculated again according to the even tempered progression (A.5). In the same way the *spdfg* set exponents are calculated. In the case of $3d$ elements an extra diffuse s auxiliary function set is added.

B Format of a platform description file

B.1 Philosophy of deMon configuration files

On most of the supported platforms, deMon comes with three sets of compilation flags (these are designated by stars in the table above). These sets are: `dbg`, `std`, and `opt`. The idea behind these options is as follows:

The `opt` (for optimized) version is intended to give maximum performance. To achieve this, this set of compilation options uses aggressive optimization flags. Compiling an `opt` version may also require vendor-optimized BLAS and LAPACK libraries (such as ESSL or SCS), which are not supplied with deMon. Often, such vendor-supplied libraries increase computational efficiency by sacrificing numerical accuracy. Because *The deMon Developers* don't have any control over the trade-off, made by the library's vendor, the `opt` version may either fail altogether, or produce wrong answers. Additionally, the location of the vendor-optimized libraries on different computers may vary. For this reason, you may have to adjust the `opt` makefile to reflect these location.

Although we make an effort to test the `opt` settings, this configuration is very sensitive to minor changes in compiler optimizations and vendor's libraries. If the `opt` version fails, please try repeating the same calculation with the `std` version, before submitting a bug report.

The `std` (for standard) version does not use any external numerical libraries, and employs conservative optimization setting. These settings are expected to result in good, but not the best possible, performance. Usually, compiling the `std` version does not require anything beyond the deMon distribution and Fortran compiler, to get it running.

Finally, the `dbg` (for debug) version allows meaningful symbolic debugging of the executable. In order to achieve high performance, the compiler will rearrange the execution order of the Fortran statements in the `opt` and `std` versions. The `dbg` version requests the compiler to generate machine code, which is as close as possible to the original Fortran source.

B.2 Structure of a platform configuration files

Platform configuration files in deMon are processes by the standard Unix `make` utility. To be accepted by `make`, these files have to follow a few simple syntactic rules:

- Lines beginning with the hash mark ("#") are treated as comments.
- A backslash character ("\") in the last column of a line indicates that a line is to be continued on the following line.
- An alphanumeric sequence, followed by the equals sign ("=") and a value indicates a *variable* assignment.
- As alphanumeric sequence, followed by a colon (":") introduces a *rule*. A rule may be followed by zero or more *actions*. Each action must consist of a tabulation character, followed by a shell command.

Variables and rules, which deMon build scripts expect to find in a platform configuration files, are summarized in Table 10.

Table 10: Structure of a platform configuration file.

`SHELL`	Name of a POSIX-compliant version of Bourne shell
`F90`	Command used to invoke Fortran-90 compiler
`FTN_FIXED`	Command line option, which selects the fixed (Fortran-77 style) source form
`FTN_FREE`	Command line option, which selects the free source form
`F90COMMON`	Common compilation flags, which will be applied to all Fortran source files. In order to compile the deMon source code correctly, you should instruct the compiler to treat `REAL` variables and constants, as well as all untyped floating-point literals, in at least 64-bit precision. On many Unix systems, the `-r8` compilation option will do this. You should also instruct the compiler to look in the deMon include directory. On many systems, `-I$(INCL)` is the right option. **Please do NOT add optimization options to this line**

`F90OPT`	"Normal" optimization flags, which should apply to most source files. On many UNIX systems, the `-O` compilation option will correspond to safe optimization setting (please consult your compiler's documentation)
`F90OPTLOW`	"Low" optimization settings, which may be required to compile a few of the source code files, which are not treated correctly at the default level. You'll be able to specify the affected files later, in `FILES_LOW`
`F90OPTHIGH`	"Extra-high" optimization settings, not suitable for most files, but required for a few of them. You'll be able to specify the affected files later, in `FILES_HIGH`
`LIBPATH`	Use this entry to specify non-standard locations of the object libraries, required to link the code. On most UNIX systems, the correct syntax is `-Lpath`. Several locations may be specified, if necessary
`LINKLIB`	Specify any additional libraries here (usually BLAS). If you include an optimized BLAS or LAPACK libraries, please make sure to exclude generic versions of these routines, supplied with deMon in `FILES_EXCLUDE`
`FILES_EXCLUDE`	Source code files (including the `.f` extension, but without the path), mentioned in this list, will not be compiled. Instead, you are supposed to provide optimized versions of these routines (see `LINKLIB` above). If you supply an optimized BLAS library (such as Atlas, or a vendor-provided library), please list the following source files in FILES_EXCLUDE: `blas1.f blas2.f dgemm.f dsyrk.f lsame.f xerbla.f`. If you link with an optimized LAPACK library (such as MKL, complib.sgimath, or LAPACK with Atlas, also list these files: `dpptrf.f dpptri.f dtptri.f dsyev.f dsyevd.f dsyevr.f dgelss.f`. Finally, if you link with a vendor-optimized EISPACK library, also exclude: `pythag.f rs.f`. All these files are located in subdirectories of `$deMon/source/math`

FILES_PLATFORM — This entry gives the list of platform-specific source code files. These source code files implement functionality, which is impossible, or difficult, to provide in a platform-independent manner. The corresponding source code files are found in the `$deMon/source/platform` directory, and its subdirectories. You need to specify the implementation for the functions `DEFLUSH` and `DETIME` (see below). The complete list of the platform-dependent implementations included with deMon is given in Table 11

FILES_LOW — The list of files (with the `.f` or `.f90` extension, but without paths), which require low optimization settings (see `F90OPTLOW` above)

FILES_HIGH — The list of files, which require extra-high optimization settings (see also `F90OPTHIGH` above

FILES_SPECIAL — The list of files, which require special treatment such as using a different compiler, or special preprocessing). Any files, mentioned on this line, will be excluded from the standard compilation process. Instead, you should provide complete compilation instructions, using Makefile rules.

ARCMD — The command, required to add (or replace) an object file to (in) an archive library. The command, will be invoked as: "`$(ARCMD) library_name object_file_name`". On most UNIX systems, you can leave the default (`ar r`) unchanged.

RANLIBCMD — The command, required to make an archive library usable by the system linker. This command will be invoked as: "`$(RANLIBCMD) library_name`" Usually, this is not required, and can be set to `RANLIBCMD=true`. However, on some UNIX systems, this should be set to `RANLIBCMD=ranlib`

`TOOLOBJ`	deMon object files, which must be included while building auxiliary tools. The usual set is: "`$(OBJ)/deflush.o $(OBJ)/detime.o`". However, if you intend to build the MAG tool, you may also need to include some of the files from the generic BLAS library.
`checkplatform:`	This make target should succeed, if this platform definition file is appropriate for the current system. The actual command line used by the driver script is: "`make -s -f platform_makefile checkplatform`". This target will typically check the operating system name, CPU model, compiler name and version. Please keep in mind that all platform-specific makefiles get invoked in this way by `make makesys`, on every platform (this is how `make makesys` determines which makefiles are appropriate)
`describeplatform:`	This target should produce a one-line description of the platform-specific makefile, on standard output. You should always put something here
`commentplatform:`	This target should produce any platform-specific installation instructions, or warnings, on standard output. If there are no specific instructions, leave this empty.

For a successful compilation, you need to provide platform-specific implementations of the following two functions:

`DETIME` This `REAL` function must return CPU time in seconds, used by the program since it started.

`DEFLUSH` This subroutine, taking one integer argument, the Fortran LUN number, must ensure that all outstanding write operations on that LUN have completed.

deMon comes with several pre-defined implementations for these routines, which are listed in Table 11. **It would be appreciated if you can send your final platform definition file to** *deMonkey@deMon-software.com***, for inclusion in future deMon distributions.**

Table 11: Platform specific functions in the shipped deMon version.

Function	Path	Description
DETIME	platform/Generic/detime-f95.F	Portable Fortran-95 routine, using CPU_TIME() intrinsic
DETIME	platform/Linux/detime.F	Use system library subroutine ETIME (Linux, HP-UX, OSF1, SunOS, IRIX, and many others)
DETIME	platform/AIX/detime.F	Use system library function MCLOCK() (IBM AIX and many others)
DETIME	platform/Generic/detime-f90.F	Portable Fortran-90 routine, using SYSTEM_CLOCK() intrinsic. This routine returns real time, rather than the CPU time, and should only be used as the last resort.
DEFLUSH	platform/Generic/qflush-POSIX.F	Use POSIX function PXFFFLUSH (IEEE 1003.9-1992, ISO/IEC 9945-1:1990)
DEFLUSH	platform/Linux/deflush.F	Use one-argument FLUSH library routine (Linux, HP-UX, OSF1, SunOS, many others)
DEFLUSH	platform/IRIX/deflush.F	Use two-argument FLUSH library routine (IRIX)
DEFLUSH	platform/AIX/deflush.F	Use FLUSH_ subroutine (IBM AIX)

C Working with the deMon Test Suite

D Global Counters, Limits and Pointers

The following tables list the most important global counter, limit and pointer variables in deMon.

Table 12: Counters.

Counters	Description	
NATOM	Number of atoms	→ IATOM, JATOM, ...
NAUX	Number of auxiliary functions	→ IAUX, JAUX, ...
NAUXSET	Number of auxiliary function sets	→ IAUXSET, JAUXSET, ...
NAUXSHL	Number of auxiliary function shells	→ IAUXSHL, JAUXSHL. ...
NCN	Degree of C axis of highest degree	→ ICN, JCN, ...
NELEC	Number of electrons	→ IELEC, JELEC, ...
NGTO	Number of primitive Gaussian functions	→ IGTO, JGTO, ...
NSEC	Number of n-fold C axes	→ ISEC, JSEC, ...
NSES	Number of n-fold S axes	→ ISES, JSES, ...
NSHL	Number of shells	→ ISHL, JSHL, ...
NSIG	Number reflection planes	→ ISIG, JSIG, ...
NSN	Degree of S axis of highest degree	→ ISN, JSN, ...
NSTO	Number of orbitals	→ ISTO, JSTO, ...

Table 13: Limits.

Limit	Description
LL(IATOM)	Number of the first orbital (STO) of atom IATOM
UL(IATOM)	Number of the last orbital (STO) of atom IATOM
LLAUX(IAUXSHL)	Number of the first auxiliary function of auxiliary function shell IAUXSHL
ULAUX(IAUXSHL)	Number of the last auxiliary function of auxiliary function shell IAUXSHL
LLAUXSET(IATOM)	Number of the first auxiliary function set of atom IATOM
ULAUXSET(IATOM)	Number of the last auxiliary function set of atom IATOM
LLAUXSHL(IAUXSET)	Number of the first auxiliary function shell of auxiliary function set IAUXSET
ULAUXSHL(IAUXSET)	Number of the last auxiliary function shell of auxiliary function set IAUXSET
LLGP(IATOM)	Number of the first grid point of atom IATOM
ULGP(IATOM)	Number of the last grid point of atom IATOM
LLGTO(ISHL)	Number of the first primitive Gaussian function of shell ISHL
ULGTO(ISHL)	Number of the last primitive Gaussian function of shell ISHL
LLSHL(IATOM)	Number of the first shell of atom IATOM
ULSHL(IATOM)	Number of the last shell of atom IATOM
LLSTO(ISHL)	Number of the first orbital of shell ISHL
ULSTO(ISHL)	Number of the last orbital of shell ISHL

Table 14: Pointers.

Pointer	Description
AUXPTR(IAUX,I)	Auxiliary function pointer
I = 1 : Origin of auxiliary function IAUX (number of atom)	
I = 2 : Auxiliary function shell	
I = 3 : AX of auxiliary function IAUX	
I = 4 : AY of auxiliary function IAUX	
I = 5 : AZ of auxiliary function IAUX	
AUXSETPTR(IAUXSET,I)	Auxiliary function set pointer
I = 1 : Origin of auxiliary function set IAUXSET (number of atom)	
I = 2 : Maximum angular quantum number L in the set IAUXSET	
AUXSHLPTR(IAUXSHL,I)	Auxiliary function shell pointer
I = 1 : Origin of auxiliary function shell IAUXSHL (number of atom)	
I = 2 : Auxiliary function set of auxiliary function shell IAUXSHL	
I = 3 : Angular quantum number L of auxiliary function shell IAUXSHL	
SHLPTR(ISHL,I)	Shell pointer
I = 1 : Origin of shell ISHL (number of atom)	
I = 2 : Main quantum number N of shell ISHL	
I = 3 : Angular quantum number L of shell ISHL	
I = 4 : Contraction of shell ISHL	
STOPTR(ISTO,I)	Orbital pointer (STO pointer)
I = 1 : Origin of orbital (number of atom)	
I = 2 : Shell of orbital ISTO	
I = 3 : AX of orbital ISTO	
I = 4 : AY of orbital ISTO	
I = 5 : AZ of orbital ISTO	

References

[1] P. Hohenberg, W. Kohn, *Phys. Rev.* **136**, B864 (1964).

[2] W. Kohn, L.J. Sham, *Phys. Rev.* **140**, A1133 (1965).

[3] R.M. Dreizler, E.K.U. Gross, *Density Functional Theory* (Springer-Verlag, Berlin, 1990).

[4] A. St-Amant, Ph.D. Thesis, Université de Montréal, 1992.

[5] M.E. Casida, C. Daul, A. Goursot, A.M. Köster, L.G.M. Pettersson, E. Proynov, A. St-Amant, D.R. Salahub, H. Duarte, N. Godbout, J. Guan, C. Jamorski, M. Leboeuf, V. Malkin, O. Malkina, F. Sim, A. Vela, deMon-KS Version 3.4, deMon Software, Montréal, 1996.

[6] M.E. Casida, C. Daul, A. Goursot, A.M. Köster, L.G.M. Pettersson, E. Proynov, A. St-Amant, D.R. Salahub, H. Duarte, N. Godbout, J. Guan, K. Hermann, C. Jamorski, M. Leboeuf, V. Malkin, O. Malkina, M. Nyberg, L. Pedocchi, F. Sim, L. Triguero, A. Vela, deMon Software, 2001.

[7] A.M. Köster, M. Krack, M. Leboeuf, B. Zimmermann, AllChem, Universität Hannover, 1998.

[8] A.M. Köster, R. Flores-Moreno, G. Geudtner, A. Goursot, T. Heine, J.U. Reveles, A. Vela, D.R. Salahub, deMon, NRC, Canada, 2003.

[9] R.G. Parr, W. Yang, *Density–Functional Theory of Atoms and Molecules* (Oxford University Press, New York, 1989).

[10] Y. Zhang, T.-S. Lee, W. Yang, J. Chem. Phys. **110**, 46 (1999).

[11] G.A. DiLabio, M.M. Hurley, P.A. Christiansen, J. Chem. Phys. **116**, 9578 (2002).

[12] Extensible Computational Chemistry Environment Basis Set Database, Version 4/05/02, Molecular Science Computing Facility, Environmental and Molecular Sciences Laboratory, Pacific Northwest Laboratory, P.O. Box 999, Richland, Washington 99352, USA.

[13] N. Godbout, D.R. Salahub, J. Andzelm, E. Wimmer, Can. J. Phys. **70**, 560, (1992).

[14] A.M. Köster, Habilitation Thesis, Universität Hannover, 1998.

[15] J. Guan, P. Duffy, J.T. Carter, D.P. Chong, K.C. Casida, M.E. Casida, M. Wrinn, J. Chem. Phys. **98**, 4753 (1993).

[16] D.P. Chong (private communication).

[17] N. Rega, M. Cossi, V. Barone, J. Chem. Phys. **105**, 11060 (1996).

[18] S. Huzinaga, J. Chem. Phys. **42**, 1293 (1965).

[19] W.J. Hehre, R.F. Stewart, J.A. Pople, J. Chem. Phys. **51**, 2657 (1969).
W.J. Hehre, R. Ditchfield, R.F. Stewart, J.A. Pople, J. Chem. Phys. **52**, 2769 (1970).

[20] A.J. Sadlej, Collection Czech. Chem. Commun. **53**, 1995 (1988).

[21] G.C. Lie, E. Clementi, J. Chem. Phys. **60**, 1275 (1974).

[22] A.J.H. Wachters, J. Chem. Phys. **52**, 1033 (1970).

[23] P.-O. Widmark, B.J. Persson, B.O. Roos, Theor. Chim. Acta **79**, 419 (1991).
R. Pou-Amérigo, M. Merchán, I. Nebot-Gil, P.-O. Widmark, B.O. Roos, Theor. Chim. Acta **92**, 149 (1995).

[24] J. Andzelm, E. Radzio, D.R. Salahub, J. Comput. Chem. **6**, 520 (1985).

[25] J. Andzelm, N. Russo, D.R. Salahub, J. Chem. Phys. **87**, 6562 (1987).

[26] `http://www.theochem.uni-stuttgart.de`

[27] H.B. Schlegel, M.J. Frisch, Int. J. Quantum Chem. **54**, 83 (1995).

[28] H.-J. Glaeske, J. Reinhold, P. Volkmer, *Quantenchemie*, Band 5, Eds. W. Haberditzl, M. Scholz, L. Zülicke (Dr. Alfred Hühtig Verlag, Heidelberg, 1987).

[29] C. Daul, Int. J. Quantum Chem. **52**, 867 (1994).

[30] P.J. Hay, J. Chem. Phys. **66**, 4377 (1977).

[31] C.C.J. Roothaan, Rev. Mod. Phys. **23**, 69 (1951).

[32] G.G. Hall, Proc. Roy. Soc. Ser. A **205**, 541 (1951).

[33] J.A. Pople, R.K. Nesbet, J. Chem. Phys. **22**, 571 (1954).

[34] C.C.J. Roothaan, Rev. Mod. Phys. **32**, 179 (1960).

[35] J.S. Binkley, J.A. Pople, P.A. Dobosh, Mol. Phys. **28**, 1423 (1974).

[36] T. Amos, L.C. Snyder, J. Chem. Phys. **41**, 1773 (1964).

[37] A.D. Bacon, M.C. Zerner, Theor. Chim. Acta **53**, 21 (1979).

[38] B.I. Dunlap, J.W.D. Connolly, J.R. Sabin, J. Chem. Phys. **71**, 4993 (1979).

[39] J.W. Mintmire, B.I. Dunlap, Phys. Rev. A **25**, 88 (1982).

[40] A.M. Köster, in preparation.

[41] J. Almlöf, K. Faegri, Jr., K. Korsell, J. Comput. Chem. **3**, 385 (1982).

[42] A.M. Köster, J. Chem. Phys., **118**, 9943 (2003).

[43] E. Cancés, J. Chem. Phys. **114**, 10616 (2001).

[44] D.R. Hartree, *The Calculation of Atomic Structures* (Wiley, New York, 1957).

[45] V.R. Saunders, I.H. Hillier, Int. J. Quantum Chem. **8**, 699 (1973).

[46] A.M. Köster, P. Calaminici, Z. Gómez, U. Reveles in *Reviews of Modern Quantum Chemistry, A Celebration of the Contribution of Robert G. Parr*, Ed. K. Sen (World Scientific Publishing Co., Singapore, 2002).

[47] P. Pulay, J. Comput. Chem. **3**, 556 (1982).

[48] B. Miehlich, A. Savin, H. Stoll, H. Preuss, Chem. Phys. Lett. **157**, 200 (1989).

[49] P.A.M. Dirac, Proc. Cambr. Phil. Soc. **26**, 376 (1930).

[50] J.P. Perdew, Y. Wang, Phys. Rev. B **33**, 8800 (1986); **34**, 7406(E) (1986).

[51] A.D. Becke, Phys. Rev. A **38**, 3098 (1988).

[52] J.P. Perdew, J.A. Chevary, S.H. Vosko, K.A. Jackson, M.R. Pederson, D.J. Singh, C. Fiolhais, Phys. Rev. B **46**, 6671 (1992).

[53] J.P. Perdew, K. Burke, M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996).

[54] Y. Zhang, W. Yang, Phys. Rev. Lett. **80**, 890 (1999).

[55] B. Hammer, L.B. Hansen, J.K. Norskov, Phys. Rev B **59**, 7413 (1999).

[56] S.H. Vosko, L. Wilk, M. Nusair, Can. J. Phys. **58**, 1200 (1980).

[57] J.P. Perdew, A. Zunger, Phys. Rev. B **23**, 5048 (1981).

[58] J.P. Perdew, Y. Wang, Phys. Rev. B **45**, 13244 (1992).

[59] J.P. Perdew, Phys. Rev. B **33**, 8822 (1986).

[60] B. Zimmermann, Ph.D. Thesis, Universität Hannover (1999).

[61] C. Lee, W. Yang, R.G. Parr, Phys. Rev. B **37**, 785 (1988).

[62] R. Colle, D. Salvetti, Theor. Chim. Acta **37**, 329 (1975).

[63] R. Colle, D. Salvetti, J. Chem. Phys. **79**, 1404 (1983).

[64] M. Krack, A.M. Köster, J. Chem. Phys. **108**, 3226 (1998).

[65] A.M. Köster, R. Flores-Moreno, J.U. Reveles, in preparation.

[66] P.M.W. Gill, B.G. Johnson, J.A. Pople, Chem. Phys. Lett. **209**, 506 (1993).

[67] V.I. Lebedev, Russian Acad. Sci. Dokl. Math. **50**, 283 (1995).

[68] A.K. Rappe, C.J. Casewit, K.S. Colwell, W.A. Goddard III, W. M. Skiff, J. Am. Chem. Soc. **114**, 10024 (1992).

[69] A.K. Rappe, W.A. Goddard III, J. Phys. Chem. **95**, 3358 (1991).

[70] M. O'Keeffe and N.E. Brese, J. Am. Chem. Soc. **113**, 3226 (1991).

[71] S.L. Mayo, B.D. Olafson, W.A. Goddard III, J. Phys. Chem. **94**, 8897 (1990).

[72] C.J. Casewit, K.S. Colwell, A.K. Rappe, J. Am. Chem. Soc. **114**, 10046 (1992).

[73] C.J. Casewit, K.S. Colwell, A.K. Rappe, J. Am. Chem. Soc. **114**, 10035 (1992).

[74] G. te Velde, E.J. Baerends, Phys. Rev. B **44**, 7888 (1991).

[75] J. Baker, A. Kessi, B. Delley, J. Chem. Phys. **105**, 192 (1996).

[76] A. Banerjee, N. Adams, J. Simons, R. Shepard, J. Phys. Chem. **89**, 52 (1985).

[77] J. Baker, J. Comput. Chem. **7**, 385 (1986).

[78] M.J.D. Powell, Math. Prog. **1**, 26 (1971).

[79] T.H. Fisher, J. Almlöf, J. Phys. Chem. **96**, 9768 (1992).

[80] R. Lindh, A. Bernhardsson, G. Karlström, P. Malmqvist, Chem. Phys. Lett. **241**, 423 (1995).

[81] J. Nichols, H. Taylor, P. Schmidt. J. Simons, J. Chem. Phys. **92**, 340 (1990).

[82] R. Fletcher, *Practical Methods of Optimization*, Second Edition (Wiley, New York, 1987).

[83] C.G. Broyden, J. Inst. Maths. Applns. **6**, 76 (1970); ibid **6**, 222 (1970).

[84] R. Fletcher, Computer J. **13**, 317 (1970).

[85] D. Goldfarb, Maths. Comp. **24**, 23 (1970).

[86] D.F. Shanno, Maths. Comp. **24**, 647 (1970).

[87] H.B. Schlegel, J. Comput. Chem. **3**, 214 (1982).

[88] W.C. Davidon, AEC Res. & Dev. Report ANL-5990 (1959).

[89] R. Fletcher, M.J.D. Powell, Computer J. **6**, 163 (1963).

[90] J.M. Bofill, J. Comp. Chem. **15**, 1 (1994).

[91] H.J.C. Berendsen, J.P.M. Postma, W.F. van Gunsteren, D. DiNola, J.R. Haak, J. Chem. Phys. **81**, 3684 (1984).

[92] R.S. Mulliken, J. Chem. Phys. **23**, 1833 (1955).

[93] P.O. Löwdin, J. Chem. Phys. **18**, 365 (1950).

[94] I. Mayer, Chem. Phys. Lett. **97**, 270 (1983).

[95] M.S. Gopinathan, K. Jug, Theor. Chim. Acta **63**, 497 (1983).

[96] R.F.W. Bader, *Atoms in Molecules, A Quantum Theory* (Clarendon, Oxford, 1990).

[97] P. Calaminici, K. Jug, A.M. Köster, J. Chem. Phys. **109**, 7756 (1998).

[98] W.F. Murphy, W. Holzer, H.J. Bernstein, Appl. Spectrosc. **23**, 211 (1969).

[99] K. Jug, F. Janetzko, A.M. Köster, J. Chem. Phys. **114**, 5472 (2001)

[100] Gonvalves, Mohallem, Chem. Phys. Lett. **280**, 378 (2003).

[101] G. Schaftenaar, Molden3.2, CAOS/CAMM Center, University of Nijmegen (1997).

[102] S. Portmann, P.F. Fluekiger, CSCS/ETHZ and CSCS/UNI Geneva (2002).

[103] R.K. Pathak, S.R. Gadre, J. Chem. Phys. **93**, 1770 (1990).

[104] A.D. Becke, K. Edgecomb, J. Chem. Phys. **92**, 5397 (1990).

[105] B. Silvi, A. Savin, Nature **371**, 683 (1994).

[106] W.E. Lorensen, H.E. Cline, Comp. Graph. **21**, 163 (1987).

[107] M. Leboeuf, A.M. Köster, K. Jug, D.R. Salahub, J. Chem. Phys. **111**, 4893 (1999).

[108] LAPACK driver routine (version 3.0) –
Univ. of Tennessee, Univ. of California Berkeley, NAG Ltd.,
Courant Institute, Argonne National Lab, and Rice University, 1999

[109] A.D. Becke, J. Chem. Phys. **88**, 2547 (1987).

[110] R.E. Stratmann, G.E. Scuseria, M.J. Frisch, Chem. Phys. Lett. **257**, 213 (1996).

[111] H.B. Jansen, P. Ross, Chem. Phys. Lett. **3**, 140 (1969).

[112] P. Fuentealba, Y. Simón-Manson, Chem. Phys. Lett. **314**, 108 (1999).

[113] K. Jug, B. Zimmermann, P. Calaminici, A.M. Köster, J. Chem. Phys. **116**, 4497 (2002).

[114] V.G. Malkin, O.L. Malkina, M.E. Casida, D.R. Salahub, J. Am. Chem. Soc. **116**, 5898 (1994).

[115] V.G. Malkin, O.L. Malkina, L.A. Eriksson, D.R. Salahub in *Theoretical and Computational Chemistry*, Eds. P. Politzer and J.M. Seminario (Elsevier, Amsterdam, 1995)

[116] W. Kutzelnigg, U. Fleischer, M. Schindler in *NMR Basic Principles and Progress*, Vol. 23 (Springer-Verlag, Heidelberg, 1990).